BUSINESS SYSTEMS WITH PUNCHED CARD DATA PROCESSING AND SYSTEM/3 MODEL 10

F. R. CRAWFORD

BUSINESS SYSTEMS WITH PUNCHED CARD DATA PROCESSING AND SYSTEM/3 MODEL 10

F. R. CRAWFORD

Prentice-Hall, Inc., Englewood Cliffs, N. J.

The author and publisher wish to acknowledge the International Business Machines Corporation for the granting of permission to reproduce the following figures: 4.1, 5.1, 5.2, 5.3, 5.8, 5.9, 5.11, 5.13, 5.15, 6.6, 6.7, 6.8, 6.10, 8.18, 8.19, 9.1, 9.2, 10.6, 10.15, 10.18, 11.1, 11.5, 11.6, 12.5, 12.9, 12.11, 13.1, 13.5, 13.6, 13.9, 13.11, 13.16, 14.1, 15.2, 17.1, 17.2, 17.5.

To my Dad, who is
always "Just delicious!"

©1973 by Prentice-Hall, Inc.
Englewood Cliffs, New Jersey

All rights reserved. No part of this book
may be reproduced in any form or by any means
without permission in writing from the publisher.

10 9 8 7 6 5 4 3 2 1

ISBN: 0-13-107698-1
Library of Congress Catalog Card Number: 72-7869
Printed in the United States of America

Prentice-Hall International, Inc., *London*
Prentice-Hall of Australia, Pty. Ltd., *Sydney*
Prentice-Hall of Canada, Ltd., *Toronto*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*

CONTENTS

PREFACE xi

1 BASIC ACCOUNTING PRINCIPLES 1

Principles of Accounting 1
Basic Accounting Formula 4
Posting to Accounts 6
Income and Expense Accounts 7
Trial Balance 8
Closing out Income and Expense Accounts 9
The Accounting Cycle 10
The Journal 10
Ledger Accounts 11
Chart of Accounts 11
Journalizing 14
Posting 14
Trial Balance 16
Closing out the Books 17
Preparing the Balance Sheet 18
The General Ledger 20
Glossary of General Accounting Terms 21
Questions and Exercises 22

2 PUNCHED CARD ACCOUNTING 23

Payroll and Labor Distribution 23
Source Documents 24
Payroll Computation 27
Payroll Writing 28
Analyze and Distribute Payroll Charges 28
Accounting Entries 28
Glossary of Payroll and Labor Distribution Terms 29
Accounts Payable 32
Glossary of Accounts Payable Terms 33
Inventory Control and Material Accounting 34
Costing 38
Accounting Entries 40
Glossary of Inventory Terms 40
Questions and Exercises 41

3 PUNCHED CARD ACCOUNTING CONTINUED 42

Order Writing and Billing 42
Reasons for Order Writing 42
Billing 44
Accounting Entries 45
Glossary of Order Writing and Billing Terms 45
Accounts Receivable 46
Accounts Receivable and the Accounting System 48
Glossary of Accounts Receivable Terms 49
Cost Accounting 50
Relationship to Accounting System 51
Glossary of Cost Accounting Terms 52
Manufacturing Control 53
Management Functions 55
Glossary of Manufacturing Terms 56
Questions and Exercises 57

4 THE PUNCHED CARD AS A UNIT RECORD 58

Data Representation for Machines 60
The 80-Column Card 63
IBM Card Code 66
Standard BCD Interchange Code 68
Card Fields 70
The 96-Column Card 72
Questions and Exercises 76

5 THE TRANSCRIPTION TO PUNCHED HOLES 77

80-Column Card Punching 78
Card Hopper 79
Punching Station 80
Stacker 80
Keyboard 81
Program Control 83
Other Features 84
The Transcription Process 87
The Machine Verification Process 88
96-Column Card Punching and Verifying 89
Punching 90
Operating Features 90
Program Control Switch 92
Program Card 93
Automatic Duplication 94
Right Adjust 94
96-Column Card Verification 94
Questions and Exercises 97

6 PUNCHED CARD DESIGN 98

Types of Punched Cards 98
Determining Card Data 102
Determining Field Size 103
Determining Data Sequence 107
Location of Data in Other Cards 107
Sequence of Data on the Source Document 109
Machine Characteristics 109
Final Card Layout 115
Questions and Exercises 116

7 PROCEDURE CONTROL 118

Record Count 121
Control Total 121
Unit Count 123
Self-Checking Number 124
Modulus 10 127
Modulus 11 129
Document Register 130
Serial Numbers for Batch Control 132

Transmittal and Route Slips 133
Questions and Exercises 134

8 SORTING AND CLASSIFICATION 135

Machine Operation 137
Sorting for Printed Reports 140
Numerical Sorting 142
Card Handling 144
Sight Checking 146
Estimating Sorting Time 146
Block Sorting 147
Alphabetic Sorting 149
Selection Switches 152
Alphabetic Shortcut Method 153
Special Sorting Devices 154
Multiple-Column Selector (IBM 82, 83) 155
IBM 5486 Card Sorter 157
Numerical Sorting: 96-Column Cards 159
Alphabetic Sorting: 96-Column Cards 160
Counter On (Special Feature) 163
Digit Select (Special Feature) 164

9 FILE MAINTENANCE 166

IBM 88 Collator 168
Sequence Checking 171
Merging Two Files 175
Merging with Selection 178
Merging with Equal Selection 183
Other File-Handling Operations 184
Matching Two Files 185
Editing 186
Comparing Two Fields in the Same Card 187
Selecting Cards by Either of Two Key Numbers 188
Selecting Zero-Balance Cards 189
Checking the Filing of Details Behind Master Cards 190
Questions and Exercises 192

10 DATA TRANSFER BY PUNCHED HOLES 193

Gangpunching 194
Interspersed Gangpunching: X-Punched Masters 195
Interspersed Gangpunching: X-Punched Details 197
Checking Interspersed Gangpunching 199
Offset Gangpunching 201
Comparing Offset Gangpunching 202
Selectors 203
Reproducing 204
Selective Reproducing 205
Combined Reproducing and Gangpunching 206
End Printing 209
Mark Sensing 210
Questions and Exercises 214

11 MACHINE CALCULATING OPERATIONS 218

IBM 521 Card Read Punch 218
IBM 604 Electronic Calculating Unit 219
604 Display Panel 221
604 Operation Planning 223
Control Panels 227
Basic Operation 227
Crossfooting: $A - B = \underline{+}T$ 227
Multiplication: $A \times B = P$ 230
Division: $A \div B = Q$ 231
Successive Calculations: $(A \times B = P) \div C = R$ 233
Sign Control 236
Approximate Square Root 237
Gangpunching 239
Questions and Exercises 239

12 REPORT PREPARATION 245

Methods of Printing 245
IBM 402 Accounting Machine 246
Card Reading 247
Control Panel 249
Control Panel Wiring 252
Detail Printing 255
Accumulating Totals 259
Program Control 263
Questions and Exercises 270

13 BASIC CONTROL PANEL WIRING 271

Counter Control 271
Multiple Selection 277
Digit Selection 280
Field Selection 282
Class Selection 284
Additional Accounting Machine Features 286
Crossfooting 286
Progressive Total Printing 287
Special Program 288
Summary Punching 288
Multiple Line Printing 292
Questions and Exercises 294

14 IBM 407 ACCOUNTING MACHINE 298

Multiple Line Read Feature 298
Storage Units 300
Tape-Controlled Carriage 302
Control Tape 303
Form Design 304
Questions and Exercises 308

15 INTRODUCTION TO COMPUTERS 310

Computer Data Representation 313
Octal Number System 317
Integer Conversion 319
Binary Number Systems 315
Fraction Conversion 321
Hexadecimal Number System 323
Questions and Exercises 326

16 COMPUTER CODES AND MAIN COMPONENTS 329

Binary Coded Decimal (BCD) 329
Standard BCD Interchange Code 331
Extended Binary Coded Decimal Interchange Code (EBCDIC) 332
American Standard Code for Information Interchange (ASCII) 333
Storage 334
Stored Program 335

Central Processing Unit 336
Functional Units 337
Counters 338
Machine Cyles 339
Program Execution 340
Questions and Exercises 342

17 ORGANIZATION OF A UNIT RECORD COMPUTER: THE IBM SYSTEM/3 MODEL 10 344

Card Input 344
Card Output 346
Printed Card Output 347
Line Printer Output 347
Processing Unit 349
Processing Unit Data Flow 351
Time Sharing 354
Interrupt 355
Instruction Format 356
Printer-Keyboard 360
Data Entry Keyboard 361
80-Column Card Compatibility 361
Questions and Exercises 362

18 IBM SYSTEM/3 MODEL 10: CARD- AND DISK-ORIENTED SYSTEMS 363

Card-Oriented Programs 363
Disk-Oriented System 368
Applications Processing Programs 375
Questions and Exercises 377

19 PROGRAM PREPARATION 379

General Considerations 380
Program Preparation 381
Program Assembly 386
Programming Systems 388
Compilers 391
COBOL 393
FORTRAN 394
Questions and Exercises 395

20 REPORT PROGRAM GENERATOR, VERSION II 397

File Description 398
Specification Sheets 398
Control or Header Card Specifications 401
File Description Specifications 404
Input Specifications 408
Questions and Exercises 432

21 RPG II CALCULATION 434

Constants 439
Use of Result Indicators 444
Operation Codes 446
Arithmetic Operations 446
Compare and Testing Operations 456
Setting Indicators 458
Branching Operations 459
Subroutine Operations 462
Other Operations 467
Questions and Exercises 468

22 RPG 11 OUTPUT-FORMAT SPECIFICATIONS AND EXAMPLES 469

Output-Format Specifications 469
Example I: Complete RPG II Program 476
Example 2: Complete RPG II Program 483
Questions and Exercises 491

INDEX 492

PREFACE

In recent years, publishers have offered an exceptional number and variety of books dealing with computers and information processing. Almost every aspect of these extensive subjects is now in print–including basic data processing concepts, hardware and software design, application of systems to problem-solving of all types, and social impact of computers.

However, there seems to have been a tendency to overlook one very important category in this field–the present and prospective users of unit record equipment. Despite this oversight, thousands of working installations of unit record equipment businesses, government departments, and military establishments throughout the world. While the total commercial consumption of punched cards continues to rise every year, it has become somewhat fashionable lately to refer to this versatile data processing medium in the past tense. Perhaps this is because the punched card was the first universally recognized symbol of Electronic Data Processing (EDP).

The need to train and educate in this field is urgent. With the introduction of the IBM System/3 Model 10, the possible application of punched cards, particularly to business problems, has greatly expanded. The new 96-column card has provided the greatest increase of speed and flexibility in the history of unit record data processing.

At the same time, punched cards have been given the versatility of completely automated processing by the stored program computer. The amount of processing that can be done in a single pass of records through the machine is no longer restricted by the relatively inflexible control panel or plugboard.

The use of punched cards has continued to expand as an input and output medium for many computers other than the System/3. New subsystems are available to read and punch cards at speeds more compatible with the processing capabilities of very large systems.

In many businesses, the accepted practices of centralization have given way to the concept of distributed processing, a policy of placing smaller capacity machines on location where equipment can be utilized by and best serve the information and accounting needs of that location. Punched card installations are ideal for this purpose. Types of card equipment have been developed that can be connected to a central computer over communication lines. Thus, the satellite installation takes care of on-the-spot, routine data handling tasks and can then communicate with the central system to report accumulated activity, maintain central files, or share peak work loads.

No substitute has yet been found for the 80-column punched card as a "turnaround" record, a record which can be prepared automatically by machine, used by people to record additional information, and can then be processed again by machine with this changed or additional data included. This concept is almost universally accepted in utility billing and credit card accounting applications.

This book reviews some basic business practices and shows how these practices can be carried out with unit record equipment. Although machine functions and capabilities are described to illustrate these practices, it is not the primary purpose of this book to prepare the student to actually operate machines. However, where pertinent, the text will offer illustrations, diagrams, and operating instructions to describe how a machine works. Actual class exercises can be carried out on machines if such equipment is available.

System/3 functional capability is described to introduce computers and to show how this system can be adapted to basic data handling and unit record manipulation procedures. Magnetic disk file function and record organization is described to show how this equipment supplements the machine's card-handling ability. An introduction to RPG II programming is included to illustrate how the computer is instructed by the programmer to perform basic tasks.

The author gratefully acknowledges the International Business Machines Corporation for the arrangements which made it possible for him to draw upon the company's extensive information sources for material for this book.

Affectionate acknowledgement is also due to my nice wife, Nancy, who helped with many hours of proofreading and with the assembly of the manuscript.

Osprey, Florida F. R. Crawford

BASIC ACCOUNTING PRINCIPLES 1

Some knowledge of basic accounting principles is helpful in understanding the purpose of electronic data processing (EDP) with unit records. These widely accepted practices are used in most commercial enterprises. For this reason, the following brief review should provide useful information for those not previously acquainted with these principles and their many related terms.

Principles of Accounting

Most American businesses use what is called the *double-entry* system of bookkeeping. As the name implies, two written entries are made in accounting records every time a transaction affects a particular business.

For example, suppose a company buys $1,000 worth of merchandise intended for retail sale and pays cash. To record this transaction, one entry in the books shows $1,000 as a reduction in the amount of cash on hand, while a second entry records a $1,000 increase in the value of inventory. This method of record keeping has become standardized so that when records are kept up to date, the financial status of a company can be accurately determined at any time. Also, the results of any company's operation can be evaluated from available financial data.

Record keeping and reporting have always been necessary functions associated with the exchange of goods and services throughout history. As early as 2300 B.C., clay and stone tablets were used to record payment for services in temples. In the famed Babylonian empire, tax levy and collection required recorded proof of individual obligations and payments. The Romans are credited with introducing the custom of budgeting government expenses.

Bookkeeping and accounting for private business developed as outgrowths of Italian commerce in the thirteenth century when Venice and Genoa became great centers of trade. In that era, as well as in present times, business ventures were financed by loans and investments of money. Special records and methods of reporting were devised to show the interests of both investors and creditors. At the same time, merchants *kept books* to record their financial dealings with both customers and employees. Double-entry records were used that began the modern accounting practices of today.

Accounting is a system of keeping the financial records of a business. The records are always stated in terms of value or money. If any distinction is to be made between bookkeeping and accounting, bookkeeping may be defined as the routine record-keeping part of accounting. Much of this routine work can be done automatically by machines, as will be shown in later chapters.

In addition to the *making* of records, accounting is concerned with the *use* of records. For example, various summaries and statements are prepared in all accounting systems. The interpretation of these statements produces information of vital importance to the management of a business. The information is also important to any others who have direct interests in the operation of the business: bankers and other creditors, stockholders, prospective investors, and government agencies.

Accountancy is the profession that practices the techniques of accounting and devises special systems for various types of businesses. A Certified Public Accountant (CPA) is one who has fulfilled all legal requirements to hold a public certificate in accountancy.

Assets are things that belong to or are possessed by a business. According to accounting practice, the value of an asset must be measurable in money. Assets are cash and property, such as land, buildings, furniture, fixtures, merchandise, and so on. Assets can also be money owed *to* a business by other businesses or individuals.

Capital is the original investment in a business, stated as value in money. Capital may be originated by the owners or by others as a portion of the assets.

Liabilities. When persons who are not the owners put assets into a business, they are known as creditors. Contribution by creditors takes place when the asset cash is loaned to a business or when merchandise or other assets are sold to it and payment is to be made at some later date. Creditors who have sold property to the business expect to get back an equivalent sum of cash at some future time, in accordance with the terms of sale.

From the time money is loaned until it is paid back, the creditors are contributors to the assets owned by the enterprise. Therefore, creditors have an interest in the business and may be considered temporary investors.

The amounts of money owed to the creditors are liabilities because a business is liable to the creditors for the sums owed.

Income. Assets are obtained by a business in two ways: (1) as direct contributions by the owners and creditors and (2) as a result of operating the business. For example, if an owner performs a customer service for cash, assets are increased by the cash amount. The increase in assets through earnings is called income or *revenue*.

A business can earn income in various ways. The greater part of income is usually from its regular activity. For example, a trading business derives income mainly from the sale of merchandise. A service business such as a garage, a laundry, or a trucking concern obtains income largely from charges for service. In addition, there may be income in the form of commissions, interest on amounts owed by customers, rent of premises let to others, and similar sources.

The increase in assets through the operation of the business by the owner is a contribution by him and so increases his interest, the capital. Thus, capital may be increased not only by direct contribution by the owner, but through the earning of income as well.

Expenses. In contrast with the increase in capital by the earning of income, capital decreases as a result of expenses incurred. The salaries paid to employees, the rent paid to the landlord, the interest on indebtedness, and the cost of telephone and other services are all examples of expenses.

In every business, there are two opposing forces at work to affect the capital: income increases capital, expenses decrease capital.

When expenses are not paid at the time they are incurred but are to be paid at some future date, the amount of such expenses is recorded as a liability. Thus, liabilities may increase as a result of contributions by creditors and by the incurring of business expense.

Net Income or Net Loss. When the total of income, or revenue, for a specific period of time exceeds the total of expenses for the same period, the result is net income for the period. Net income causes an increase in capital.

When the total of expenses for a period exceeds the total of income, the result is a net loss for the period. A net loss results in a decrease in capital.

Cash and Accrual. Accounting is practiced on two bases: cash and accrual.

On a cash basis, income is considered as earned only when payment is actually received. Expenses are taken into consideration only when actually paid for.

On an accrual basis, income is considered earned when goods are sold or services performed, even though payment may not have been received. Similarly, expenses are taken into consideration when they are incurred, although they may not be paid.

Basic Accounting Formula

The basic double-entry accounting formula evolved from the earlier Italian system late in the fifteenth century. It was decided then that everything owned by a business should be called *property*. This idea can be expressed as an equation

$$\text{Property} = \text{ownership}$$

If the accounting terms explained in the previous section are substituted, the equation becomes:

$$\text{Assets} = \text{capital}$$

To illustrate this axiom, assume that a business starts to operate with $10,000 in cash. To record this situation, the first entries to the accounting records would be made thus:

$$\text{Assets} = \text{Capital}$$
$$\$10{,}000 = \$10{,}000$$

This equation must always balance; that is, assets must always equal capital.

Now assume that in the course of its operation the business buys land worth $3,000 and pays in cash. Two entries must be made to the accounts to record this transaction and to keep the original equation in balance.

	Assets	=	capital
	$10,000	=	$10,000
Land	+3,000		
Cash	−3,000		
	$10,000	=	$10,000

Next, the business buys $2,000 worth of inventory for cash:

	Assets	=	capital
	$10,000	=	$10,000
Land	+3,000		
Cash	−3,000		
Inventory	+2,000		
Cash	−2,000		
	$10,000	=	$10,000

The business now buys a truck for $4,000. But rather than pay cash, it is decided to pay at a later date. The business now assumes a liability, or an *account payable.* This transaction is entered as follows:

	Assets	—	liabilities	=	capital
	$10,000				$10,000
Land	+3,000				
Cash	−3,000				
Inventory	+2,000				
Cash	−2,000				
Truck	+4,000		−$4,000 (accounts payable)		
	$14,000		−$4,000	=	$10,000

Note that the equation is still balanced. However, a liability category has been added. Because a liability is something a business owes, this amount must be deducted from the value of the assets to show the true equity of the ownership account. To do this, the basic accounting formula is stated as:

Assets — liabilities = capital

or

Assets = liabilities + capital

The above formula shows that a business owns its liabilities as well as its capital. Capital can also be expressed as net worth. Therefore, another form of the basic formula is:

Assets = liabilities + net worth

Posting to Accounts

Accounting transactions could be listed as just shown for an entire accounting period. However, in this form it would be rather difficult to see what transactions affected each type of account, such as cash on hand, accounts payable, accounts receivable, and so on. It would also be difficult to determine the status of each account; for example, how much money is owed or due the business at any time.

Much more convenient access to individual accounts can be maintained by the arrangement shown schematically in Figure 1.1.

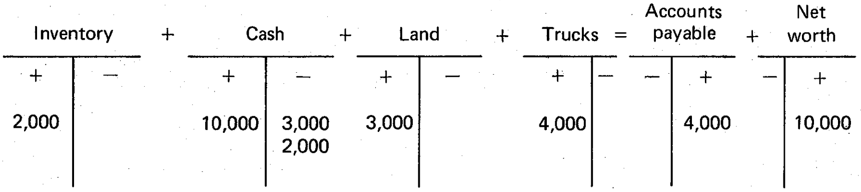


Figure 1.1 Formation of "T" accounts.

Notice that each account is set up in the form of a "T." Asset accounts are considered as being to the left of the equals sign while liabilities and net worth accounts are to the right. The arrangement makes posting considerably easier, especially when done manually. Two rules are always followed:

1. For asset accounts, those to the *left* of the equals sign in Figure 1.1, posting to the left of the T increases the account. Posting to the right decreases it.

2. For accounts payable and net worth accounts, those to the *right* of the equals sign, posting to the left of the T decreases the account. Posting to the right increases it.

In accounting, posting to the left side of an account is called a *debit* entry; posting to the right side is a *credit* entry. For all practical purposes, the terms debit and credit mean left and right postings. The rules may be summarized as follows:

1. For asset accounts, increases are recorded by debits; decreases by credits.

2. For liability and ownership accounts, increases are recorded by credits; decreases by debits.

Income and Expense Accounts

In a strictly commercial environment, a company engages in business by selling either service or merchandise for a profit. Sales increase the net worth account on the ownership side of the basic equation.

However, in the conduct of every business, there are expenses. These must be subtracted from net worth. Or, as shown graphically in Figure 1.2, net worth is increased or decreased by the amount of the difference (profit or loss) between sales and expenses.

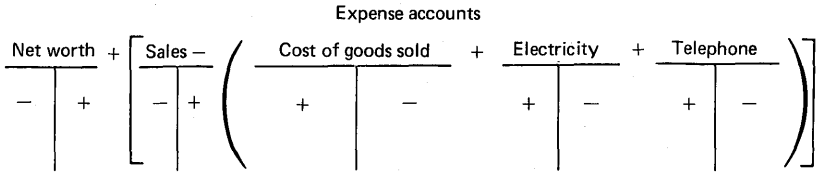


Figure 1.2 Relationship of net worth to sales and expenses.

Rather than attempt to calculate the effect on net worth of each expense as it occurs, the double-entry system provides for setting up individual accounts for each category of expense. This arrangement is shown in Figure 1.3. At the end of an accounting period, these accounts can be *closed out* to a profit and loss account which is then posted to net worth. Thus, for any accounting period, it can be readily determined whether the business is operating at a profit or loss.

Note that the signs for the expense accounts shown in Figure 1.3 are reversed from the normal sign indication on the ownership side of the equation. That is, the debits are plus while the credits are minus. Mathematically, when the accounts are closed out to profit and loss, the signs will be changed by the minus sign in front of the parenthesis.

To illustrate this process, assume the following transactions:

1. The business makes a $300 sale for which it receives cash. The cost of goods sold is $150.

2. The business receives a bill of $25 for electric power and pays the bill in cash.

3. A telephone bill is received for $50 but not paid now.

To account for these transactions, the following postings are made:

1. Cash is increased (debited) $300; sales is increased (credited) $300.
Cost of goods sold is increased (debited) $150; inventory is reduced (credited) $150.

2. Electricity is increased (debited) $25; cash is decreased (credited) $25.

3. Telephone is increased (debited) $50; accounts payable is increased (credited) $50.

Figure 1.3 shows the postings to all accounts as they would appear in the basic equation.

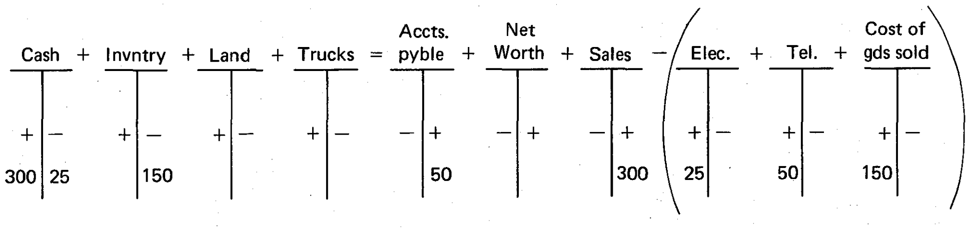

Cash + Invntry + Land + Trucks = Accts. pyble + Net Worth + Sales − (Elec. + Tel. + Cost of gds sold)
Cash: + | − ; 300 | 25
Invntry: + | − ; | 150
Land: + | −
Trucks: + | −
Accts. pyble: − | + ; | 50
Net Worth: − | +
Sales: − | + ; | 300
Elec.: + | − ; 25 |
Tel.: + | − ; 50 |
Cost of gds sold: + | − ; 150 |

Figure 1.3 Double entry for expenses.

Trial Balance

Assume now that the transactions shown so far complete the business activity for an accounting period. The next step is to prove the accuracy of the double-entry postings; that is, prove that the debits equal the credits. This is accomplished by taking a *trial balance*.

First, individual account balances are calculated by adding the debit and credit columns of each account and then subtracting the smaller total from the larger. The trial balance is prepared by listing and totaling the resulting balances, debit or credit, as shown in Figure 1.4.

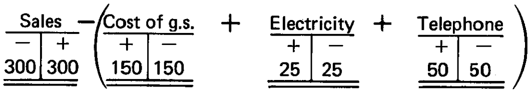
Trial balance		
Account	Debits	Credits
Cash	$ 5,275	$
Inventory	1,850	
Land	3,000	
Trucks	4,000	
Accounts payable		$ 4,000
		50
Net worth		10,000
Sales		300
Cost of good sold	150	
Electricity	25	
Telephone	50	
	$ 14,350	$ 14,350

Figure 1.4 Trial balance.

After the trial balance is taken, adjusting entries can be made for income earned or for expenses incurred where no entries were recorded. For example, the accounting period might end in the middle of a pay period. In this case, an entry is made for pay earned by the employees but not yet paid. An adjusted trial balance is then prepared.

Closing Out Income and Expense Accounts

The next step is to close out the income and expense accounts as shown in Figure 1.5. The sales account is *closed out* by debiting it $300 and crediting the profit and loss account for the corresponding amount. Each expense account is closed by crediting



	Profit & loss account	
	(Expense)	(Income)
Cost of g. s.	150	300 Sales
Elec.	25	
Telephone	50	
	225	300
	* 75	
	300	300

*The difference between the income and the expense items is the profit or the loss.

Figure 1.5 Profit and loss account.

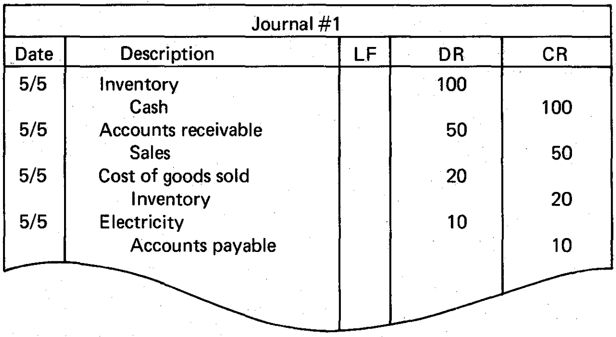
with the proper amount and debiting the profit and loss account with the corresponding amount. The difference between total income and expense is profit or loss. The difference is posted to the net worth account. Profit credits the account while a loss debits net worth.

The Accounting Cycle

An example of a typical accounting cycle described in the following section will further explain the concepts described thus far.

The Journal

As business transactions occur they are normally posted in a *journal*, the first entry in the company records. The journal can be kept in the form of either a book or a sheet. It serves three primary purposes (Figure 1.6):



Journal #1

Date	Description	LF	DR	CR
5/5	Inventory		100	
	Cash			100
5/5	Accounts receivable		50	
	Sales			50
5/5	Cost of goods sold		20	
	Inventory			20
5/5	Electricity		10	
	Accounts payable			10

Figure 1.6 Accounting journal.

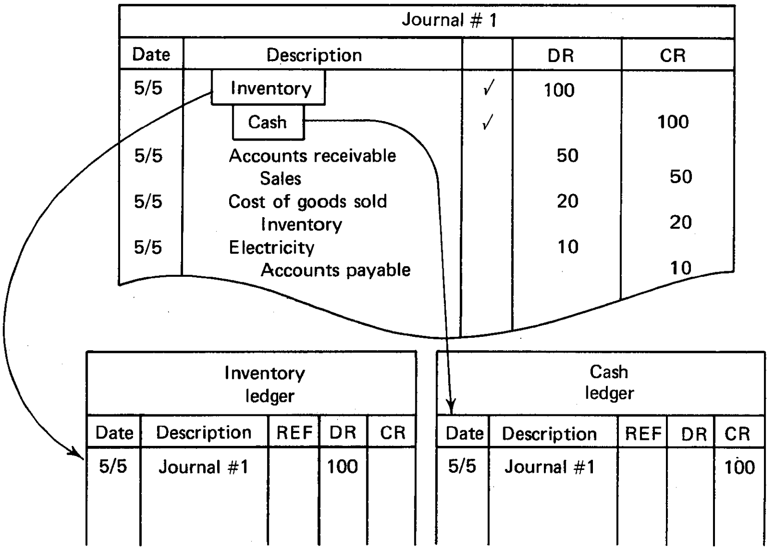
1. Pertinent facts about all transactions are shown in the sequence in which they occur. This is often very important for reference and auditing purposes.

2. Offsetting debit and credit entries are both shown for each transaction with a note of explanation. Thus, an identifying record of all transactions is kept in one location.

3. Possibility of error is reduced by requiring both debit and credit entries to be shown. If transactions were posted directly to a ledger, there would be a chance that a debit or credit entry might be omitted.

Ledger Accounts

At this point, all transactions have been listed in chronological order. Therefore, it is difficult to isolate postings by any particular category for reference (for example, entries affecting only inventory or accounts payable). The next step is to post from the journal to individual T accounts kept in a *ledger*. It is then possible to examine all transactions affecting a particular account during any specific period. This procedure is shown schematically in Figure 1.7.



Journal # 1

Date	Description		DR	CR
5/5	Inventory	√	100	
	Cash	√		100
5/5	Accounts receivable		50	
	Sales			50
5/5	Cost of goods sold		20	
	Inventory			20
5/5	Electricity		10	
	Accounts payable			10

Inventory ledger

Date	Description	REF	DR	CR
5/5	Journal #1		100	

Cash ledger

Date	Description	REF	DR	CR
5/5	Journal #1			100

Figure 1.7 Posting from journal to ledger.

Chart of Accounts

A *chart of accounts* lists all of the accounts to which the company wishes to post transactions. When posting is completed for a given period, the accounts will be used to prepare financial statements of the business. Posting to the various accounts is often referred to as the distribution of amounts. In many businesses, the number of such transactions runs into the thousands each month. Processing requires a great deal of effort and expense. A sample chart of typical accounts follows:

Cash
Capital stock
Land
Equipment
Cost of goods sold
Notes payable
Net worth
Inventory
Profit and Loss
Sales
Accounts payable
Transportation
Accounts receivable
Surplus
Buildings
Equipment
Utility expenses

To classify these accounts, it would be helpful first of all to assign them to broad categories, then break these categories down into finer classifications, as follows:

Assets
Liabilities
Net worth
Income
Expense

When the accounts are properly placed under the appropriate major classification, numbers are assigned to facilitate the recording of entries.

100			Assets
	110		Current assets
		111	Cash on hand and in banks
		112	Accounts receivable
		113	Reserve for bad debts
		114	Notes receivable
		115	Marketable securities
		116	Inventory
	120		Fixed assets
		121	Buildings
		122	Depreciation reserve for buildings
		123	Land
		124	Equipment and machinery
		125	Depreciation reserve for equipment and machinery
200			Liabilities
	210		Current liabilities
		211	Accounts payable
	220		Long term liabilities
		221	Notes payable
300			Net worth
		331	Capital stock
		332	Surplus
400			Operating income and expense
	410		Income
		411	Sales

420		Expense
	421	Cost of goods sold
	422	Fuel
	423	Electricity
	424	Telephone
430		Profit and loss

Income and expense accounts are often referred to as *temporary* accounts because they reflect business activity over a fairly short period of time and are closed out at the end of the accounting period. Listed below are examples of income and expense accounts.

Income	*Expenses*
Sales	Cost of goods sold
Interest income	Rent
Other income	Insurance
	Electricity
	Fuel

For purposes of illustration, the following transactions of a company are shown posted to a journal, then transferred to the proper ledger accounts. Assume that the company—

1. purchases $100 worth of inventory and pays for it in cash;
2. makes a sale of $50 (the cost of goods sold is $20);
3. receives an electric bill for $10;
4. receives $50 in cash for a prior sale;
5. makes a sale of $150 (the cost of goods sold is $75);
6. buys inventory valued at $5,000;
7. receives a fuel bill for $15;
8. pays a $10 electric bill;
9. pays the $5,000 due for the inventory purchase;
10. makes a sale of $75 (the cost of goods sold is $35);
11. pays a $15 fuel bill;
12. receives a $20 phone bill.

Journalizing

As these transactions occur, they are recorded in the journal. First, each transaction is examined to determine the way in which the various asset, liability, and net worth accounts are affected. The date of the transaction is entered in the date column. Next, the amount of the transaction is entered to the accounts to be debited and credited. If required, a note of explanation is included. A blank line is left after each journal entry.

The posted journal with the above transactions appears in Figure 1.8. These entries have not yet been posted to the ledger.

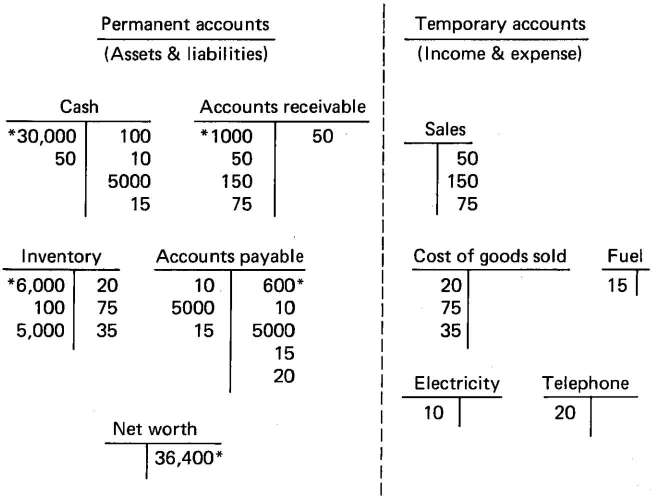
Journal

Date		Account debited	Account credited	L.F.	DR	CR
May	5	Finished goods inventory			100	
			Cash			100
	5	A/R			50	
			Sales			50
	5	Cost of goods sold			20	
			Finished goods inventory			20
	5	Electric			10	
			A/P			10
	5	Cash			50	
			A/R			50
	8	A/R			150	
			Sales			150
	8	Cost of goods sold			75	
			Finished goods inventory			75
	8	Finished goods inventory			5000	
			A/P			5000
	8	Fuel			15	
			A/P			15
	8	A/P			10	
			Cash			10
	8	A/P			5000	
			Cash			5000
	8	A/R			75	
			Sales			75
	8	Cost of goods sold			35	
			Finished goods inventory			35
	8	A/P			15	
			Cash			15
	8	Telephone			20	
			A/P			20

Figure 1.8 Journalizing daily transactions.

Posting

The next step is posting from the journal to the ledger accounts. T accounts are used in the illustration (Figure 1.9), with debits posted to the left side and credits to the right. Notice that there are certain balance amounts already in the permanent accounts shown. These are marked with asterisks. The items represent money left in the accounts at the close of the previous accounting period.



Permanent accounts (Assets & liabilities)

Cash	
*30,000	100
50	10
	5000
	15

Accounts receivable	
*1000	50
50	
150	
75	

Inventory	
*6,000	20
100	75
5,000	35

Accounts payable	
10	600*
5000	10
15	5000
	15
	20

Net worth	
	36,400*

Temporary accounts (Income & expense)

Sales	
	50
	150
	75

Cost of goods sold	
20	
75	
35	

Fuel	
15	

Electricity	
10	

Telephone	
20	

Figure 1.9 Posting from journal to ledger accounts.

1. The first entry debits inventory $100, increasing the value of the asset account. The offsetting entry credits cash $100, reducing cash by this amount.

2. For the $50 sale, accounts receivable is debited to increase assets. Sales is credited for the same amount. The merchandise cost the company $20. Because this is an expense, cost of goods sold is debited $20. The inventory account is credited with the same amount.

3. The electricity account is debited $10 to increase expense. Because the bill is not yet paid, accounts payable is credited $10 to increase liability.

4. For the $50 payment received from a prior sale, the asset cash is increased by debiting this account $50. An offsetting entry of $50 is credited to accounts receivable, reducing this account.

5. For the $150 sale, accounts receivable is debited to increase assets; sales is credited to increase income. Since the cost of goods sold is $75, a debit is made by that amount. The offsetting entry is a credit to inventory of $75, reducing this asset account.

6. Inventory is debited $5,000 for the value of purchases. Accounts payable is credited for $5,000 because the company did not pay cash.

7. For the $15 fuel bill, the fuel account is debited $15; accounts payable is also credited for the same amount.

8. As the $10 electricity bill is paid, accounts payable is debited $10 to reduce liability. The offsetting credit reduces the asset cash account.

9. When the company pays for the $5,000 inventory purchase, accounts payable is debited $5,000 to reduce liability. The offsetting entry credits cash for $5,000.

10. For each sale not made for cash, four entries are made: (a) debit $75 to increase accounts receivable and (b) credit sales for the same amount to increase income; (c) debit cost of goods sold for $35 and (d) reduce the value of inventory by crediting the inventory account for the same amount.

11. When the outstanding $15 fuel bill is paid, accounts payable is debited for $15 to reduce a liability account and cash is credited the same amount.

12. The $20 telephone bill is also an expense. The telephone account is therefore debited and the accounts payable credited with the offsetting amount of $20.

Trial Balance

To assure the accuracy of all postings to the account ledgers, a listing is made for each account showing the debit and credit entries for all transactions posted during the current accounting period. This listing is called a *trial balance*. If the period's transactions are properly posted, the total of the debits will equal the total of credits.

The trial balance shown in Figure 1.10 is made up of the transactions that were just posted to the T accounts.

For purposes of illustration, the trial balance shows all transactions that have occurred during the period. This is slightly different from common accounting practice which provides for first determining account balances by adding the debit and credit columns of each account, subtracting the smaller sum from the larger, and then taking the trial balance by listing and totaling the resultant balances.

Trial balance		
Account	Debit	Credit
Sales		$ 50
		150
		75
Cost of goods sold	$ 20	
	75	
	35	
Fuel	15	
Electricity	10	
Telephone	20	
Cash	50	100
		10
		5,000
		15
Accounts receivable	50	50
	150	
	75	
Inventory	100	20
	5,000	75
		35
Accounts payable	10	10
	5,000	5,000
	15	15
		20
	$10,625	$10,625

Figure 1.10 Final trial balance.

Closing Out the Books

If it is assumed that these are all the transactions for an accounting period, it is now a major concern of management to determine what profit, if any, was made. This involves *closing out the books*. To arrive at the profit or loss for a corporation during a given period, the temporary accounts (income and expense) are used. Again, for purposes of illustration, the statements will be prepared by working directly from the T accounts. In actual practice, statements are prepared by working with the account balances developed on the trial balance (Figure 1.11).

The first step is to add all the figures in the sales account. The total of $275, when debited to the sales T account, closes it out. The offsetting entry is a credit to the profit and loss account.

The next area of concern is the cost of goods sold account. The total is $130. By crediting this amount and debiting to the profit and loss (P & L) account, the cost of goods sold account is closed out. The fuel, electricity, and telephone accounts are handled in the same manner.

The debits in the P & L account represent all the expenses for the accounting period. The difference between the debits and credits is therefore the profit or surplus for the accounting period. In this example, the total is $100. To reflect the transfer of profit to the net worth section, the P & L account is debited $100 and the net worth account is credited $100.

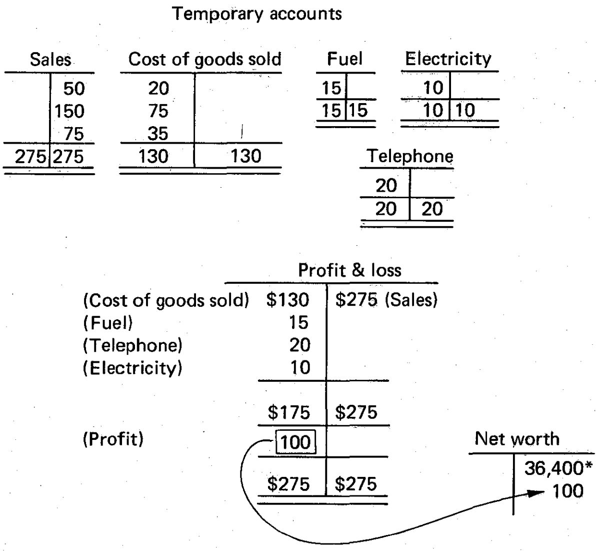


Figure 1.11 Temporary accounts.

The profit and loss account is also called the *income statement*. The statement shows management the profit made by the company during the accounting period. In this example, the profit or surplus represents an increase of $100 in the net worth of the business.

Preparing the Balance Sheet

The next step in the closing procedure is to post the asset and liability accounts to the balance sheet to show the assets, liabilities, and net worth of the company. To do this, individual accounts are closed out by first adding the debits and credits for each account. Then the smaller sum is subtracted from the larger to determine the balance for that particular account.

For example, all the debits in the cash account are added for a total of $30,050. All the credits are added for a total of $5,125. The smaller credit balance is subtracted from the debit balance to produce a remainder of $24,925 debit. Notice that this figure and all other totals of the various accounts are marked with two asterisks. This is to indicate that these amounts are the closing balances for the current account period, as well as the opening balances for the next period.

Permanent accounts

Cash	
* 30,000	100
50	10
	5,000
	15
** 24,925	

Accounts receivable	
* 1,000	50
50	
150	
75	
** 1,225	

Inventory	
* 6,000	20
100	75
5,000	35
** 10,970	

Accounts payable	
10	600 *
5,000	10
15	5,000
	15
	20
	620 **

Net worth	
	36,400 *
	100
	36,500 **

Balance sheet

Cash	$24,925	Accounts payable	$ 620
Accts. receivable	1,225		
Inventory	10,970	Net worth	36,500
	$37,120		$37,120

Figure 1.12 Permanent accounts and balance sheet.

The debit balance for the accounts receivable account is $1,225.

For the inventory account, the beginning debit balance of $6,000 is added to the other entries, $100 and $5,000, to arrive at total debits of $11,100. The credits are totaled by adding the entries $20, $75, and $35. The difference between the total debits and total credits is $10,970. This is the closing balance for the closing period and the opening balance for the next.

The accounts payable credit balance is $620.

The net worth account has a starting balance of $36,400. When the surplus amount of $100 is added for the current period, total net worth is increased to $36,500.

Cash, accounts receivable, and inventory (all asset accounts) are added to arrive at a total of $37,120. The entries are made on the left side of the balance sheet as shown in Figure 1.12. On the right side of the balance sheet, accounts payable of $620 (liabilities to others) and $36,500 (liability to stockholders) are totaled for

$37,120. Thus, assets equal liabilities plus net worth, as stated in the basic accounting formula.

The General Ledger

When punched card data processing procedures are used, the general ledger entries are in the form of punched cards rather than in the form of manual postings on a ledger. Thus, at the end of the accounting period, either just before or after the completion of the income statement and balance sheet, a machine-listed general ledger report is prepared. This report is a summary of all accounts and their balances. In many cases, it also shows the opening balance of each account and the total debits and credits posted to the account for the period.

In order to provide a complete historical record in the form of general ledger accounts, the totals (debit and credit) of the activity in the temporary income and expense accounts are shown, even though these accounts were previously closed out to the P & L account and the profit from it, in turn, was closed out

General ledger

Acct no	Acct title	Opening balance	Debits	Crdts	Closing balance
	Assets				
110	Current assets				
111	Cash	$30,000	$ 50	$5,125	$24,925
112	Accounts recvbl	1,000	275	50	1,225
116	Inventory	6,000	5,100	130	10,970
	Total assets				$37,120
	Liabilities				
210	Current liabilities				
211	Accounts payable	600	5,025	5,045	620
300	Net worth	36,400		100	36,500
	Total liabilities				37,120
	Operating income & expense				
410	Income				
411	Sales			275	
	Total income				275
420	Expense				
421	Cost of goods sold		130		
422	Fuel		15		
423	Electricity		10		
424	Telephone		20		
	Total expense				175
430	Profit & loss				100

Figure 1.13 General ledger.

to the net worth account. A sample of the general ledger is shown in Figure 1.13.

Although the above presentation has been somewhat simplified, the basic steps apply to most accounting procedures. A number of application areas will be summarized in chapters 2 and 3 to illustrate how the procedures apply.

Glossary of General Accounting Terms

Asset Something owned by a company or business, such as cash and accounts receivable.

Balance sheet A statement showing in total all the assets *owned* and all the liabilities *owed* by the company.

Chart of accounts A listing of all accounts to which a company will post transactions.

Credit An amount posted to an account to decrease it, if an asset, or increase it, if a liability. Credits are posted to the right side of the account.

Debit An amount posted to an account to increase it, if an asset, or decrease it, if a liability. Debits are posted to the left side of an account.

Distribution The allocation of amounts to desired accounts.

Fixed asset Something owned by a company intended for long-term use and not acquired for resale. The investment in a fixed asset will be recovered through future operations of the business.

Income statement The financial report that shows the profit or loss made by a company during a particular accounting period.

Journal The book or sheet to which transactions are first posted in chronological sequence.

Ledger The sheet to which entries from the journal are posted. These sheets are maintained by individual account classification, for example, cash and inventory.

Liability Money owed by a business.

Net worth The stockholders' equity in the business. It represents the difference between assets and liabilities.

T account Another name for ledger account. The T is made by the intersection of the horizontal line under the account name with the vertical line separating debits from credits.

Trial balance The listing and totaling of all accounts to insure accuracy. Debits must equal credits.

Voucher A business form that shows all the details of a transaction and authorizes the entry of that transaction into accounting records.

Questions and Exercises

1. What is the main advantage of the *double-entry* bookkeeping system?
2. Identify the distinction between accounting and bookkeeping.
3. Express the basic accounting formula.
4. Summarize the rules of debit and credit posting to accounts.
5. What is the purpose of the trial balance? Journal?
6. List some examples from a typical chart of accounts.
7. Post the following transactions to a journal in the sequence given, showing accounts debited and credited.

   Sell $50; cost of goods is $24.
   Receive cleaning bill of $16.
   Sell $200; cost of goods is $145.
   Buy inventory worth $2,000.
   Receive $29 from previous sale.
   Sell $60; cost of goods is $40.
   Receive utility bill of $30.
   Sell $90; cost of goods is $75.
   Pay $2,000 for inventory purchase.
   Receive fuel bill of $120.
   Sell $250; cost of goods is $190.
   Receive phone bill of $80.
   Receive $1,000 from previous sale.

8. Post from the above journal to a ledger.
9. Prepare a trial balance to check posting.
10. Close out books to determine profit or loss for the period. Assume same previous balances to accounts as shown in Figure 1.12.

PUNCHED CARD ACCOUNTING 2

Chapter 1 outlined general accounting practices more or less common to commercial enterprises of all types. It can be shown that the methods of carrying out these practices have also been gradually standardized throughout the business community, so that many procedures are similar for such activities as payroll, cost distribution, and inventory control. Individual procedures may vary considerably in detail, depending upon the business, but the general approach is usually consistent with established practice. In addition, federal, state, and local regulations for revenue and tax reporting make it necessary to conform to legal requirements for providing information to government. Thus, considerable uniformity of systems has been mandated by law.

Chapters 2 and 3 present an overview of basic accounting procedures which have historically furnished the majority of unit record applications. A glossary of terms follows each explanation. Many of the terms will be used later in the text to relate these accounting procedures to actual data processing steps as they are described.

Payroll and Labor Distribution

The term *payroll* refers to procedures for recording time worked by a number of employees and to the calculation and payment of the resultant wages due. Earnings are reported to each employee, the employer, and to federal and state governments as required by law. Payroll procedures include the calculation and withholding of federal income and social security taxes. Local, state, and city taxes may also be involved, depending upon the place of employment and the residence of the employee.

Payroll is probably the most extensively mechanized procedure of all modern business practices. It is widely accepted as a standard application of computers and punched card data processing. Payroll is a required procedure wherever people exchange their time, skills and services for money, not only in business but in governments, institutions, and military services as well.

All employers must compute employee earnings and maintain records from which wage and tax reports can be prepared. This is often a voluminous record-handling process involving the effort and expense of many people. In many enterprises, labor is the largest and most significant single expense of doing business. If not properly controlled, this cost can be a major cause of an unprofitable operation or inefficient use of public funds.

Employers must be sensitive to good employer–employee relations. Therefore, it is extremely important for wages to be computed correctly and paid promptly at the close of the established pay period. At the same time, the system must be designed to respond readily to changes in individual situations: wage changes, promotions, location changes, and so on. The operation of the payroll procedure must reflect the employer's concern for prompt and accurate remuneration of his employees' services.

Labor distribution is a procedure closely associated with payroll. It concerns the charging, or distribution, of labor costs to the particular job to which the employee is assigned—by number, account, department, location, et cetera.

Source Documents

Basic steps required to prepare payroll and labor distribution remain essentially constant, no matter what type or method of payment is used or what tools are applied to perform the record-keeping function. The first step is to create and maintain source documents.

1. The personnel or *master payroll record* includes such typical information as employee name, social security number, and number of tax exemptions. These basic facts are required for

income tax purposes. In addition, many companies assign an employee serial number for internal identification within their own organization.

The personnel record usually includes other pertinent statistics about the employee, including age, marital status, sex, skill or job code, date employed, assigned location and shift, and so on. These facts may not only be needed for tax purposes, but for matters of company policy and union contracts. For example, the employee's age affects insurance and retirement benefits. Date employed establishes seniority. Skill or job code may determine work assignment and location.

The record usually identifies the type of payroll involved. An employee may be salaried, that is, paid regularly by the week, month, or other period under contract or agreement. Or, an employee may work strictly by the hour at an agreed-upon hourly rate. Either salary or hourly wages may be affected by overtime, bonuses, commissions, or production incentives of various kinds. The master record states the rate of pay for hourly employees or the base amount of salary per pay period.

The master record can show the employee's work location, assigned department or division, and even the method of payment if desired (i.e., by check or in cash).

2. The *attendance record*, usually a time or clock card, is a document on which an employee registers time at work. A time clock is usually provided for this purpose at a location conveniently near where the employee arrives and leaves at the beginning and end of the work day. The card clearly identifies the employee, usually by both man number and name. When the card is inserted in the clock, the time of day is automatically registered in the proper position.

Employees are often required to sign the attendance record at the end of the pay period to signify that the document is a true record of the time worked. The employee's immediate supervisor may also endorse the document to testify that the time worked is also agreed upon by the company management and that the employee is, in fact, properly entitled to compensation.

The time card provides a record of employee absence or lateness, with the reason for any deviation from regular working hours. The card is a vital record in payroll procedure and may serve as the basis for settlement of any dispute between employer and employee over wages due.

3. *Labor distribution* records are prepared. An employee may spend time on a number of different jobs during the pay period, or he may be assigned a station on a production or continuous process line. In any case, the time and resultant wages must be charged to the particular job or jobs at which the employee worked. That is, the labor cost must be *distributed* to the proper accounts.

Usually, the creation of labor distribution records is the function of a *timekeeper*, an employee charged by management with the responsibility of keeping records of how employee time is spent.

The timekeeper often monitors the recording of employee time on the clock cards and serves as auditor of the attendance records. Total time distributed for any employee must equal his clock time for the period.

4. *Deduction authorization*. In addition to automatically withholding income taxes, an employer can deduct other amounts from wages when authorized to do so by an individual employee. For example, regular payments may be deducted for savings accounts, credit unions, government savings bonds, stock purchase options, hospitalization and personal insurance, union dues, and so on. Deductions may be either fixed amounts or a percentage of gross pay. Or, authorized amounts may be varied in other ways from one pay period to the next.

An employee may repay an employer by payroll deduction for advances against wages or for special equipment needed to perform his job but not furnished by the company, such as uniforms, safety shoes, safety glasses, tools, automobile or truck, et cetera. A company can be required by law to collect or *garnishee* a portion of an employee's wages to satisfy creditors.

Records of deduction authorizations must be maintained from pay period to pay period and inserted in the payroll procedure at the proper time. Many deduction procedures are supplementary data processing applications of considerable volume, involving extensive record-handling facilities.

A deduction register is prepared at some point in the payroll run to show an itemized listing of all deductions taken for the pay period.

5. *Personnel records*. In many instances there are associated personnel records that serve as the basis for payroll wage data. Rate of pay is normally authorized by management through personnel representatives. Pay scales can be established in many ways:

by collective bargaining, by a merit system, civil service scale, military rating, professional standing, seniority, yearly contract within a wage structure, and many others. In any case, the rate of individual compensation is basic data normally received by the payroll department from some other area within the organization.

Conversely, payroll records are used often in the determination of wage rates which can be affected by such factors as performance ratings, attendance records, production results, and job assignment.

Payroll Computation

When all source records are properly prepared and ready at the end of the pay period, the payroll computation can begin. This phase will include at least the following basic steps:

1. Controls are established to assure that all attendance records and other source data are received and properly authorized. For example, the payroll department accepts a number of time cards from each department as received from the timekeeper. For hourly employees, a control total should be immediately set up by the total number of hours worked by each department. In all subsequent runs, a check back to these totals can verify that all employees have been accounted for at various points during the procedure. For salaried employees, a man count can be set up which will serve the same purpose as a control on hours worked.

2. Rates of pay or salary amounts are related to each employee's record of time worked as authorized by the personnel department.

3. Gross earnings are calculated by multiplying hourly rates times hours worked. Earnings may also include additional pay for overtime, incentive, bonuses, commissions, and other supplementary earnings.

4. Federal and local income taxes are calculated for each employee.

5. Social security taxes are calculated.

6. Miscellaneous deductions are taken, according to the employee's authorization.

7. Net pay is calculated according to the following formula:

$$\text{Gross} - \text{withholding tax} - \text{social security tax}$$
$$- \text{miscellaneous deductions} = \text{net pay}$$

Payroll Writing

A completed payroll includes a number of printing operations:

1. A payroll register is prepared showing all amounts for all employees on the payroll. At the same time, year-to-date earnings and taxes are summarized. Totals must balance against preestablished controls.

2. Individual earnings and deduction statements are prepared for each employee.

3. Checks or cash envelopes are prepared.

4. A deduction register is run and corresponding deduction accounting is completed.

5. Periodically, federal and local tax reports are printed.

6. Bank account reconciliation is performed.

7. Other management and control reports are made available as required by the business.

Analyze and Distribute Payroll Charges

Payroll charges are distributed to orders and accounts covering manufacturing and operating costs. Expenses can be analyzed by comparing controllable units with budgets or standards.

Accounting Entries

The relationship of payroll cost to the general accounts may be illustrated by the examples shown in Figure 2.1.

a. As payroll expense is incurred, it is charged to the payroll control account. However, until such time as wages are paid, the company has incurred a liability, so the accrued payroll account is credited with the $500 payroll cost.

b. Payroll expense must be recovered by charging it to work in process. In Figure 2.1, it is assumed that $400 of payroll cost was spent for direct labor. This can be charged directly to a work-in-process account.

c. It is also assumed that $100 of payroll expense was for indirect labor, that is, for foremen, timekeepers, and so on. This labor is not chargeable to a work-in-process account. Instead it is charged to an actual burden account. At the end of the accounting period, all such costs will be distributed back to the various production departments and then worked into the cost of work in process.

d. At the time wages are paid, the liability reflected up to this point is discharged into the accrued payroll account. This has the effect of lessening the cash position, as the credit to cash indicates.

The net result of this series of transactions is to add the cost of labor to work in process, while lessening the cash position. This temporary loss should be retrieved when the product is sold.

Payroll expense

(A) 500	400 (B)
	100 (C)

Work in process

(B) 400	

Accrued payroll

(D) 500	500 (A)

Actual burden

(C) 100	

Cash

	500 (D)

Figure 2.1 Relationship of payroll costs to general accounts.

Glossary of Payroll and Labor Distribution Terms

Accrued payroll The liability for earnings due to employees since the end of the previous pay period.

Actual cost The purchase or contract price of goods and services used in manufacturing or other operations involving cost accounting records, as opposed to estimated or standard costs.

Advance An amount of money paid to an employee before the customary time of payment for services.

Attendance card (or *clock card*) A card showing the time an employee spends at his place of employment.

Base rate The hourly rate of pay for an employee.

Bonus Increased earnings or some material reward in excess of regular earnings granted to an employee for accomplishing results equal to or in excess of a set standard, or for working less desirable hours or under less desirable conditions than others.

Burden That part of the cost of producing goods and services which cannot be directly associated with the items produced.

Clock number A number assigned to an employee and placed on his time card for identification purposes.

Clock station The location of a time recorder where the time cards for employees are placed.

Continuous job card A card containing time worked and other pertinent facts concerning the work of one employee on one job order, process, department, or operation during successive days within a pay period.

Current earnings card The unit record used to accumulate an employee's earnings and other pertinent data for a pay period.

Daily time card A card containing attendance time and other pertinent facts concerning one employee for one day.

Day shift A period of working time for employees which begins in the morning and ends sometime in the afternoon.

Deduction card A card containing pertinent facts concerning an amount to be deducted from an employee's pay for such things as contributions, insurance premiums, bond purchases, dues, company purchases and advances.

Deduction register A list showing the amounts deducted from employees' earnings, and the reason for each deduction.

Denominating cash payroll Determining the number of bills and coins of each denomination required to make up each employee's pay with a minimum number of bills and coins.

Direct labor cost Employees' earnings that are directly applicable to a job order or process.

Distribution Assignment of payroll data to the various accounts affected.

Earnings record A record of employee earnings and taxes, both for some specific pay period and cumulative to date.

Earnings statement A report given an employee, usually at the time he is paid, showing the factors pertinent to his earnings, taxes, deductions, and net pay.

Employee or man number A number assigned to an employee for identification purposes. Usually same as clock number.

Gang job card A card containing all pertinent facts about the work of several employees working as a unit on one job order, process, department, or operation.

Gross earnings card A punched card containing an employee's total earnings before taxes or deductions.

Group bonus plan A bonus plan whereby the earnings of each employee are increased when the production by a group of employees with whom he works is in excess of the set standard.

Idle time The time during which an employee or machine is available for work but is not working.

Indirect labor The work of those employees who contribute to the overall production of goods and services, but whose work is not directly applied to the product.

Individual job card A card containing all pertinent facts about the work of one employee in one day on one job order, process, department, or operation.

Job order A written authorization to perform a specified task.

Master personnel card For automatic data processing, a punched card containing information about an employee's education, marital status, tax exemptions, age, sex, company and occupational seniority, salary or hourly rate, and other factors needed for company records.

Master rate card A card containing information that sets the rate of pay for the person or thing identified. An employee's rate is expressed in terms of money, usually rate per hour. In production or manufacturing, the rate may be expressed in terms of quantity, for example a standard number of pieces per hour.

Payroll advice A memorandum prepared for the payroll department authorizing or giving notice of some change in the status of an employee.

Payroll deduction An amount authorized to be subtracted from an employee's earnings before payment is made by the employer.

Payroll master card A card with constant or semiconstant data concerning an employee: name, serial number, social security number, occupation, rate of pay.

Payroll register A detailed list prepared for each pay period of all employees paid for that period. The register usually contains the same information shown on the employee's payroll check and earnings statement.

Reconciliation of bank statement A comparison of the bank statement with the cash records of the business to see whether they are in agreement. Adjustments must be made for any discrepancies.

Standard cost An accounting system which uses predetermined costs of labor and material required to produce goods and services.

Time sheet A list containing the names or man numbers of a group of employees, showing the time worked by each employee for a day, a week, or a pay period.

Unit cost The purchase price or manufacturing cost of an item.

Variance The difference between actual and standard measurement.

Work-in-process file A file of punched cards containing labor distribution and material requisition cards for commodities which are being manufactured.

Accounts Payable

Accounts payable is the term applied to the procedure of recording and accounting for the amount of money a business owes for goods and services received from other businesses or individuals. The procedure is established to record what is owed, to pay indebtedness promptly and thus receive allowed discounts, and to keep management informed regarding the amount and purpose of expenditures.

Most businesses follow a general procedure which involves the following steps:

1. Requisitions for goods or services are approved by designated management authority.

2. Formal purchase orders are forwarded to suppliers as authorization to ship merchandise or perform services.

3. Goods or services are received.

4. Invoices and purchase orders are audited for agreement while actual receipt of goods or services is verified.

5. Invoices are placed in a file for payment on due date.

6. Checks and check register are prepared on due date.

7. Distribution reports are prepared showing expense by account number for later use in general ledger and cost accounting.

8. Various management reports are prepared, for example, to show purchases by individual vendor.

Accounts payable has a close relationship to many other accounting applications. The source data for accounts payable is usually entered on a voucher which consolidates all the data necessary for payments and distributions. After processing, the data developed provides input for cost, plant and equipment, inventory and material, and general ledger accounting. Various accounts are affected by accounts payable transactions. For example:

a. As goods are received, the invoice is audited and approved for payment on the due date. The accounts payable account is credited $50 to increase liability, as shown in the example in Figure 2.2 Inventory is debited $50 to increase the asset account.

b. Payment is made on the due date. If by the terms of the invoice, the business is allowed a cash discount of 10%, accounts payable is debited $50 to decrease the liability account. At the same time, cash is credited $49 and discount is credited $1. Discount is considered other income because the business did not have to pay out $50 but only $49. The discount entry balances the double entry, making debits equal credits.

Accounts payable	
(B) 50.00	50.00 (A)

Inventory	
(A) 50.00	

Cash	
	49.00 (B)

Discount	
	1.00 (B)

Figure 2.2 Posting for amount received on account.

Glossary of Accounts Payable Terms

Accounts payable distribution reports Distribution of paid expenses by account number according to the chart of accounts.

Accounts payable voucher The document used to consolidate all data necessary for payment. Sources of data are purchase orders, vendors' invoices, and material receipts.

Allowances Concessions or reductions against an invoice brought about by varying conditions and reasons, and agreed to between the purchaser and the seller.

Anticipation discount An amount which the customer may subtract if he chooses to pay an invoice before its due date. The amount is usually a fixed percent of the invoice total, the percent agreed upon between buyer and seller.

Apron A form attached to vendors' invoices with space for management approval, vendor code number, voucher number, account distribution code, and amount.

Cash disbursement register A listing of paid vendors' invoices.

Credit An increase in indebtedness to a vendor's account.

Debit A decrease in indebtedness to a vendor's account.

Due date The date on which, according to the terms and date on the invoice, payment must be made.

Invoice register A listing of vendors' invoices approved for payment.

Returns Items of merchandise sent back to a vendor for one reason or another. Credit is given for its corresponding value or original charge.

Short shipment A shipment that proves to contain less than the quantity billed.

Sticker Like an apron, but gummed so that it can be pasted to either the face or the back of the vendor's invoice.

Trade discount Discount earned by special type of business. For example, an automobile parts manufacturer will give a discount to a wholesaler but charge the catalog price to a garage.

Trial balance A listing of outstanding liabilities by vendor.

Inventory Control and Material Accounting

Inventory control is a procedure to determine the amount and type of parts, supplies, and finished goods that will most effectively protect the production, sales, and financial requirements of a business. Material accounting is concerned with the financial or money control of inventory.

Much of the capital of a business is likely to be invested in inventory. Therefore, management should have answers to the following questions for effective control:

What is in stock?
What is on order?
Will it meet requirements?
When should we reorder?
Is it too much?
How much should we reorder?

What is in Stock?

There are two basic methods of keeping inventory:

1. *Physical Count.* Each item in inventory is manually counted to determine how much is in stock. However, when inventory is taken periodically, the amount of stock on hand can be known accurately only after the count has been completed. In

most industries, physical inventory is taken no more than once or twice a year. With large inventories, it is impractical to count in shorter periods because of the excessive time and expense involved.

2. *Perpetual Inventory.* A record is kept of each item in stock to be controlled. The primary record is the inventory balance card, updated as each tansaction occurs. When items are received, the quantity is added to the record of balance on hand. When items are withdrawn from stock, the quantity is subtracted from the record of what is on hand. Thus, the record reflects the stock balance at all times, as long as the transactions are accurately posted to the balance card. The formula is:

Old balance + receipts — issues = on hand

With the perpetual book inventory system, a physical inventory is taken at least once a year as a verification of the accumulated balances. Where very large inventories are maintained, a continual manual check of selected items may be made, for example, for items with exceptionally high value or volume of transactions. In addition, items may be selected at random for counting as a further check on the functioning of the system.

What is on Order?

Besides maintaining records which show inventory on hand, it is also necessary to control the material on order. This is true both for purchased items and items to be manufactured by the business. This information is required because any decision to obtain additional inventory must be based upon a knowledge of any open orders. The basic formula now becomes:

Old balance + receipts — issues = on hand
On hand + on order = available

Will It Meet Requirements?

Once the quantity of all items on hand is known, it must be determined whether this amount is sufficient to meet orders and thus prevent loss of sales and customers. Also, a minimum allowable balance should be decided upon for each item of stock so that the item will not be completely depleted before more can be obtained. The formula can be expanded to include these factors.

Old balance + receipts = issues
On hand − minimum balance = reorder
On hand + on order − planned issues = available

When Should We Reorder?

To provide assurance against out-of-stock conditions, two reorder methods can be used:

1. Stock clerks can watch the amount of stock on the shelves to see that it does not get too low. Generally this method is relatively inefficient, especially when there are many different items.

2. Minimums can be established and recorded on stock balance records to signal when items should be reordered. Factors used to determine minimums are—

   a. past usage, such as average sales for a specified period;
   b. the total time, usually called *lead time*, needed to place the order, process it, and receive delivery;
   c. a margin of safety that should be maintained in case the ordered amount does not arrive on time or in case usage is greater than anticipated.

The standard formula for establishing minimum order points is:

$$\text{Minimum} = \frac{(\text{total sales for a specified period} \times \text{lead time})}{\text{specified period}} + \text{safety factor}$$

For example, assume that usage of a given item for a four-week period is 10,000 pieces and that lead time to obtain an additional supply is two weeks. Substituting in the above formula, the minimum balance is determined to 5,000 pieces, or exactly two weeks supply. This is exactly the time it takes to have more items in stock. As a safety factor, therefore, it might be considered that an extra week's supply, or 2,500 pieces, might be sufficient. Thus, the total minimumm balance to be kept in stock becomes 5,000 plus 2,500 pieces or 7,500.

Is It Too Much?

Because the yearly cost of maintaining inventory can be 25% or more of the total dollar value of the inventory, it is important not to stock too much.

The cost elements that make up the commonly quoted 25% figure are:

Obsolescence	10%
Interest	6%
Depreciation	5%
Miscellaneous	4%.
Total	25%

These factors are variable, depending upon the type of inventory involved. For example, an inventory of bottled liquors does not become obsolete nor is it often subject to depreciation. On the other hand, breakage costs might very well run higher than 4%. An inventory of wearing apparel, however, is clearly subject to drastic obsolescence from style changes alone. Current economic conditions affect other costs, such as interest and warehouse maintenance.

How Much Should We Reorder?

In the retail and wholesale industries, a standard quantity is usually ordered when the minimum order point is reached. Among the factors that determine this quantity are:

1. *Vendor Discount for Size of Order.* It is usually true that the larger the quantity ordered, the lower the unit purchase price. However, a vendor may have established his most economical lot size and offer discounts for orders in increments of this lot quantity. For example, 5,000 bushels of grain may fill one freight car and is therefore the most economical lot quantity to handle and ship. An order for 5,250 bushels may require a great deal of special handling and result in a much higher unit price for the extra 250 bushels.

2. *Storage and Handling Costs.* This cost is largely determined by the size and weight of the item relative to cost. For example, the handling and storage cost for major appliances is a sig-

nificant factor in the cost of distributing them to the customer. Probably the same costs for diamonds is insignificant, relative to their value. The perishable nature of a product may also be a factor in determining the best order quantity, for example fresh fruits and produce.

3. *Order Processing Costs.* This cost may be particularly significant where elaborate procedures are followed to process each order. The cost of preparing the order may exceed the cost of the item to be stocked. This is particularly true for small quantities of hardware items such as nuts, bolts, screws, and washers.

4. *Danger of Obsolescence.* When obsolescence is a factor in inventory, considerable judgment must be exercised in the maintenance of stock balances. In some cases obsolescence can be anticipated by the introduction of new models, techniques, and products. Changes in style, consumer tastes, and weather conditions cannot always be readily evaluated in terms of their effect on inventory.

Costing

The preceding discussion was mainly concerned with the quantity control of inventory. The cost aspect must also be considered. Costing will provide answers to two additional questions:

> What is the value of the inventory on hand?
> How much should we cost each item when
> it is issued or sold?

In dealing with *finished goods* inventory, there are three predominant methods of costing:

1. *Average Cost.* The total dollar value of an item in stock is divided by the total quantity. For example, if there are 100 pieces in inventory and their total value is $100.00, the average unit cost is $1.00. However, if 100 more pieces were purchased for $120.00, the average unit cost would be $1.10, the result of dividing $220.00 by 200 pieces.

2. *FIFO* (*First-In, First-Out*). The material acquired first is used first and is costed accordingly. This method demands that each purchased lot be accounted for separately. For instance, assume there are three 100-piece lots of an item in inventory with

unit costs of $1.00, $1.20, and $1.30, respectively. If 110 pieces are sold, the costing of the sale is:

100 pieces for $1.00 each = $100.00
10 pieces for $1.20 each = $ 12.00
$112.00

Therefore, remaining in inventory are

90 pieces for $1.20 each
100 pieces for $1.30 each

The FIFO costing method is especially applicable in those industries where receipts into inventory are in relatively large quantities and where each lot can be conveniently costed individually.

3. *LIFO (Last-In, First-Out)*. As the name implies, material sold is costed according to the lot most recently received. When that lot is exhausted, the cost of the next most recent lot is used. This requires that each incoming shipment be maintained as a distinct lot in keeping the records.

Under the LIFO method, if a sale of 110 pieces of the above item were made, the costing procedure would be:

100 pieces for $1.30 = $130.00
10 pieces for $1.20 = $ 12.00
$142.00

The LIFO method has been used more and more widely in recent years. In periods of rising costs, the method tends to reduce the amount of stated profit and resultant taxes.

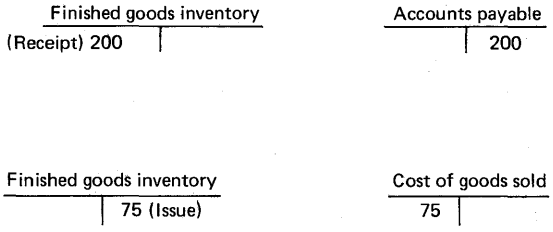


Figure 2.3 Finished goods and accounts payable posting.

Accounting Entries

The value of an inventory must be reflected in the accounting books of a company. When merchandise is received, the inventory account is increased. Since inventory is an asset, a debit entry will increase that account, as shown in Figure 2.3.

When a sale is made, the value of inventory is decreased by the cost of the items withdrawn. Therefore, a sale of merchandise costing $75 credits the finished goods inventory and debits cost of goods sold by the same amount.

Glossary of Inventory Terms

Available The quantity of stock on hand, plus the quantity on order, minus the quantity reserved for specific purposes.

Average cost The cost of each piece of an item in inventory, arrived at by dividing the total dollar value of the inventory by the number of pieces in inventory.

Balance card A punched card on which the quantity of stock on hand is recorded. The card may also include other quantities, such as maximum and minimum, reservations, issues, receipts, and available stock.

Cycle inventory taking A variation of complete physical inventory, and often used to augment it, in which checking of actual stock against stock records is done on a rotating basis, a portion of the inventory at a time.

EOQ (economical order quantity) Use primarily in manufacturing organizations, EOQ is the result of a formula which incorporates the pertinent factors required to determine the most economical number of pieces to be ordered or manufactured when stock reaches the reorder level.

FIFO (first-in, first-out) A method of costing goods issued using the cost of the goods received first as the cost of the goods issued first.

Finished stock Items or products which have been manufactured or purchased and are ready for sale or use without further manufacturing or processing.

Issues The amount of inventory released for production or sale.

Lead time The number of hours or days necessary to place an order, process it, and receive the material in inventory.

LIFO (last-in, first-out) A method of costing goods issued, using the cost of goods received last as the cost of goods issued first.

Perpetual book inventory Maintaining an up-to-date record of all inventory balances.

Physical count An actual count of all pieces of stock in an inventory.

Receipts Merchandise or stock that is received in inventory.

Requisition Written authorization to release items from stock.

Reservation A means of assuring that certain required quantities of stock will remain available for some future need.

Standard order quantity A fixed number of pieces ordered when the reorder level for an item is reached.

Stock on hand The quantity of any item or commodity actually located in stockroom and available for use or issue.

Stock status report A report prepared by various means to show by item the quantity on hand, quantity on order, quantity reserved or set up as minimum inventory and thus available for issue, and any other pertinent data.

Transaction register A list of transactions, such as issues, receipts, and adjustments, which affect the balance of material on hand.

Questions and Exercises

1. Define payroll and labor distribution procedures.
2. List five primary payroll and labor distribution records used in nearly all procedures.
3. Outline some important controls that can be established over payroll procedures. What purposes are served by these controls?
4. Review the accounting entries showing the relationship of payroll cost to general accounts.
5. What is the purpose of an accounts payable procedure?
6. Show some possible types of accounting controls for an accounts payable procedure. State the purpose of each.
7. Give reasons why an efficient accounts payable procedure is good business practice.
8. Describe two basic methods of keeping inventory.
9. Determine the minimum requirements for an item in stock if 200 pieces are used every five weeks. Lead time is three weeks.
10. List factors that determine proper order quantity.

3 PUNCHED CARD ACCOUNTING CONTINUED

Order Writing and Billing

No company offering commodities or services for sale can operate without receiving orders. When an order is obtained, a company's first objective is to disburse to the buyer the commodities or services ordered in the fastest, most economical manner possible and then to be paid accordingly.

Following sections explain how an order is handled after it is received by the seller and the means taken to request payment from the customer.

Reasons for Order Writing

An order is often received in handwritten form, written either by the customer or an authorized sales representative. After an order is received, the factory or warehouse must be notified to produce or ship whatever is requested by the customer.

The source document used for this purpose can be the order itself. However, when orders originate from salesmen or from many different customers, the document usually does not follow any standard form. Such orders are apt to be illegible, confusing in nomenclature, incomplete in description, and irregular in appearance. If this document is processed, it can result in incorrect shipments, lost time, and a generally unsatisfactory operation.

To avoid this possibility, many companies rewrite orders received from customers or sales representatives. By established procedure, a new document is prepared which describes the goods or services ordered by the customer.

A complete customer's order should include ship-to and sold-to name and address, date, order number, quantity ordered, description of merchandise, and such special information as shipping, packing, and marking instructions.

An order writing procedure offers the following advantages:

1. A standard form convenient to use
2. Assured legibility
3. Correct and complete description
4. Correct pricing and packing

In addition, many by-products are obtained as follows:

1. Acknowledgment to customers that their orders have been received. A company with an order writing procedure will generally send the customer the original copy of the order when it is written.

2. Shipping list or packing list. Some companies require a list before commodities are assembled and packed for shipment. The list shows the details concerning shipping units, that is, the number of cans, cases, carloads, etc. for each item.

3. Shipping labels, usually attached to each shipping unit and showing the name and address of the customer.

4. A bill of lading, to authorize a rail or motor transportation company to move a commodity from vendor to customer. The document shows the destination, route, freight class, and gross weight.

5. Back order control. When a customer's order cannot be filled, or can only be partially filled, the vendor may prepare a back order to advise the customer what items have to be sent at some later date. The material to eventually fill the back order is either on order or will have to be ordered.

6. Order analysis. This is one of the more important benefits obtained by management from an order writing procedure. It usually consists of a daily tabulation of all orders received, showing totals by commodity. The report gives management an up-to-

date picture of how sales are moving. In addition, the order analysis report should give ample warning so that production and/or purchasing can be geared to handle the volume of orders on hand. Order analysis can be combined with tabulations of shipments to report on efficiency in filling orders. Periodic reviews of this order-shipment report can greatly influence production and purchasing policies, produce a more efficient and economical operation, and help keep back orders to a minimum.

Billing

After goods are shipped or services rendered, the vendor must notify the customer and state the charges. A document is needed to show the goods or services purchased by the buyer, the name of the buyer, the seller's name, and the terms.

A typical invoice for goods sold is composed of essentially three types of information:

1. Heading information. This includes customer name and address, customer number, terms, branch and warehouse, salesman, and how shipped.

2. Miscellaneous data. Information can include customer's order number, invoice number, date, and special shipping instructions.

3. Body. Commodity information is listed here—quantity and pack, description, unit price, and selling price with discount if applicable.

Much of the data used in a billing operation is repetitive because many companies normally deal with the same customers over and over again and the products sold remain fairly constant.

Billing can be a by-product of order writing where order writing cards are reused, or it may be a separate procedure. Some companies produce an order and keep the original copy on file. When items are shipped to a customer they pull the original order; write in quantity shipped; perform the extension of quantity times unit price; total the invoice; fill in such information as invoice number, invoice date, and shipping instructions; and then use the order as an invoice to be sent to the customer.

With any billing application there is always a method of handling credit memorandums. When merchandise is returned to a vendor for any legitimate reason, a credit memo is issued to the customer along with a check for the credited amount or a statement reducing the indebtedness. The preparation of credit memos parallels that of invoices and usually takes much the same form. By the same token, a company will have to issue debit memos if a customer has been charged less than the correct amount.

Accounting Entries

After an invoice is prepared, the following accounts are affected:

1. Accounts receivable is debited for the amount of an invoice total; sales is credited for the same amount.

2. Inventory is credited with the total cost of the items withdrawn; cost of goods sold is debited for the same amount. (See Figure 3.1.)

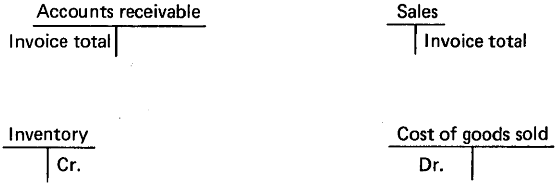


Figure 3.1 Accounting entries for billing.

The sale of any commodity is reflected by many accounting functions. For example, totals appearing on an invoice and recorded in an accounts receivable register are shown on the customer statement of account and are included in a monthly aged trial balance of accounts receivable. The trial balance is a report showing a list of customers accounts which are past due and the amount owed by date.

In addition, the stock record of quantity on hand must be reduced by the quantity of the individual item sold. This transaction is reflected in the stock status report. Finally, the desired sales analysis and a number of sales reports (by salesman, by customer, by product, by geographical location) all originate from individual sales transactions. Because of this chain reaction, accuracy is very important.

Glossary of Order Writing and Billing Terms

Acknowledgment Notice to the customer that his order has been received by the vendor.

Back order An order prepared to cover items which cannot be included in the original shipment, but which will be sent when available.

Bill of lading Document which the vendor must produce to give a transportation company authority to move a commodity from vendor to customer. The document must show destination, routing, freight class, and gross weight.

Credit memo Document issued to a customer detailing merchandise he has returned from a vendor, or other adjustments for which cash is owed by the vendor to the customer.

Debit memo Document representing a charge to the customer for corrections, additions, and special or other unusual charges.

Description Details required to identify a given merchandise item or commodity.

Heading information Customer data used in preparing the shipping order and invoice heading.

Invoice Document that describes the commodity purchased and indicates name of buyer, name of seller, terms, and amount of sale.

Miscellaneous data Variable invoice details that are peculiar to a given sale and cannot be predetermined.

Order Document describing goods or services required by a customer.

Order analysis Classification and summarization of facts about products or customers from orders not yet filled.

Ship to Heading which calls for the name and address to which merchandise will be delivered.

Shipping label Form usually attached to each container in a shipment, showing the name and address of the customer who will receive the shipment.

Shipping list List of commodities packed or assembled for shipment, showing details concerning units to be shipped.

Sold to Heading which calls for the name and address of the buyer.

Source document Original record of a transaction.

Terms Condition on which the sale is made; for example, cash, net 30 days, cash on delivery, etc.

Vendor An alternate term for seller.

Accounts Receivable

It has been estimated that 90% of American business is conducted on a credit basis. Only 10% of the customers in the United States pay for their commodities and services in cash. This widespread acceptance of credit places an additional burden upon nearly every type of business to establish and maintain a strict control of customer indebtedness. This record-keeping task is known as accounts receivable. As an accounting procedure, it is readily adaptable to unit-record data processing methods.

Accounts receivable appears on a company's statement of condition as an asset, and may be sold if desired. Many companies do this to maintain working capital. For example, when a customer buys an appliance from the local dealer, a contract may be signed arranging to pay for the purchase in regular monthly installments. The dealer commonly sells this contract to a bank or finance company at a discount thereby converting this account receivable to cash almost immediately. The bank or finance company also takes over the record-keeping task from the dealer and collects the indebtedness from the customer.

Customers have widely varied paying habits. If a business is to realize a satisfactory conversion of accounts receivable to cash, it must know both the credit status of prospective customers and the paying habits of current customers.

The general objectives of the accounts receivable procedure are—

1. to collect money owed in the shortest possible period of time;

2. to reduce loss from bad debts;

3. to maintain customer goodwill through prompt and accurate updating of payment records.

Regardless of the accounts receivable system used, the following overall steps must be taken:

1. The customer's account is established. This may involve a credit check by either the business or a credit agency. The customer's identity is established with name and address, type of account, and usually a credit limit.

2. A record is set up within the procedure which will be used to record transactions to the account as they occur. Transactions are purchases, payments, returns, and adjustments.

3. The accuracy of entries is controlled so that the business knows at all times how much the customer owes and when payment is due.

4. Total accounts receivable control is maintained and a collection procedure is established.

Accounts Receivable and the Accounting System

It has been established that accounts receivable is an asset that, in most cases, can be readily converted to cash. The following example shows how typical accounts receivable transactions are handled on a company's books. The T account postings are shown in Figure 3.2.

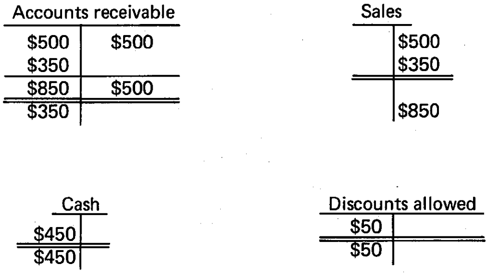


Figure 3.2 Posting for accounts receivable.

1. A sale of $500 is made. Sales is credited for $500, reflecting income in that amount. Accounts receivable is debited to establish an asset of $500.

2. The customer pays his bill and is entitled to a 10% discount. Since he has discharged his obligation to the business, accounts receivable is credited $500. The payment increases cash, therefore the cash account is debited $450. The difference between cash and accounts receivable is represented by the $50 discount. This represents an expense of doing business which is reflected on the books by debiting discounts allowed, an expense account.

3. A sale of $350 is made.

If these three transactions represented an entire month's business, the effect on the financial statements at the end of the month would be as follows:

1. Accounts receivable is closed out to the balance sheet as a current asset.

2. Cash of $450 is closed out to the balance sheet as a current asset.

3. Sales of $850 are closed out to the profit and loss statement as income.

4. Discount allowed of $50 is closed out to the profit and loss statement as expense.

Glossary of Accounts Receivable Terms

Accounts receivable ledger The overall record of customer indebtedness. The system used might be a ledger book, a file of individual customer ledger cards, a file of IBM punched cards, a magnetic tape file, or other means.

Aged trial balance Same as a trial balance except that open items are listed in separate groups according to age; for example, all items billed up to thirty days ago, all items billed 30 and 60 days ago, and all items billed more than 90 days ago. This report enables management to quickly analyze problem accounts and take appropriate action.

Allowance An adjustment to a customer's bill, generally authorizing additional credit.

Cashier Generally, the person assigned the responsibility of accounting for and controlling customer payments, either by cash or check.

Control sheet Document, generally posted daily with summary totals from other reports, and used to prove that all entries affecting the accounts receivable ledger have been properly posted and that the accounts receivable ledger itself is correct.

Credit memo Document authorizing credit to a customer because of damaged merchandise, a billing error, returned goods, etc.

Current balance Amount owed by a customer at any given time.

Cycle statement writing The practice of subdividing the entire accounts receivable file into alphabetic groups, and rendering statements to each group on different dates. This eliminates the peak loads that occur when all statements are rendered at one time.

Debit memo Document increasing the original amount of an invoice because of a billing error, shipment error, etc.

Entry date Date on which an invoice, payment, or adjustment is entered into the accounts receivable file.

Invoice register Daily listing of invoice totals by customer. It also shows indicative and classification data such as date, invoice number, terms, etc.

Journal voucher Internal document used to make miscellaneous entries to accounts receivable.

Open item A bill that has not been paid.

Partial payment A customer remittance covering only part of a bill.

Remittance statement Document prepared by the customer, and enclosed with his check, to describe the invoice(s) being paid. It generally shows invoice number(s), invoice amount(s), and discount(s), and is used by the vendor to properly credit the customer's account.

Return Merchandise returned by the customer to the vendor for credit.

Statement Document periodically sent by the vendor to the customer (frequently at month's end) which shows the total amount owed to the vendor on unpaid bills.

Trial balance Periodic listing of all open items to prove that the accounts receivable ledger is in balance with the control sheet.

Cost Accounting

Cost accounting is concerned with the identification and accumulation of the costs of manufacturing and marketing a product. This section will deal only with the cost of manufacturing, which consists of:

1. The cost of raw material, parts, hardware, etc., used in making the article
2. The cost of labor used directly to manufacture the article
3. A fair share of the manufacturing overhead, i.e.,
   a. indirect labor (e.g., supervisors, clerks, accountants, and management in general)
   b. indirect material (e.g., paint, wire, wrapping and packing materials)
   c. power and heat, gas, compressed air, etc.
   d. depreciation of plant and equipment

This can be stated in formula form:

Cost = direct material + direct labor + apportioned overhead

By accumulating unit costs for products, management can analyze the profitability of these products in comparison with selling price and can recognize inefficient production methods or the need to revise the selling price.

When records are kept of costs by operation and job, this information can be used for pricing new products, for estimating the cost of future products, and for determining economical ordering quantities. By keeping track of costs by department, it is possible to use this information to establish budgets for coming periods of operation.

By comparing current cost figures to preestablished estimates or standards, inefficient areas can be spotted for management to take corrective action.

The objectives of cost accounting may be summarized as follows:

1. Accumulation and identification of costs of production for each job completed

2. Accumulation of costs for each area or department

3. Making the proper accounting entries for the work flow through the plant

4. Preparation of management reports

Relationship to Accounting System

As material moves from raw material inventory into work in process and ultimately into finished goods inventory, associated accounting entries are made to indicate the work flow. In like manner, entries must be made to record the expending of labor and other expenses as they are incurred. The following four transactions illustrate this:

1. As labor is expended in the production departments, the value of work in process is increased by the amount of this labor. The debit to direct labor work in process accomplishes this increase as shown in Figure 3.3. An offsetting credit to payroll increases the accumulation of the amount owed to employees, a liability account.

2. As material is entered into the production cycle, it also increases the value of the work in process. A debit to direct material work in process reflects this increase as also shown in Figure 3.3. The offsetting credit to stores indicates a reduction in raw material supply, an asset account.

3. As expenses such as heat and light are incurred, they also contribute to the production effort and increase the value of work in process. This is called overhead or *burden*, the third cost factor. A debit to manufacturing overhead work in process increases this

account to show the incurred overhead expenses. The offsetting credit to accounts payable increases the amount owed to suppliers. Other sources of overhead would be indirect labor and indirect material.

4. As products are completed and leave the production floor to be placed in stock and held for sale, corresponding accounting entries are made. The debit to finished goods inventory reflects an increase in this account. The offsetting credit to the three work in process accounts indicates a reduction in the work in process.

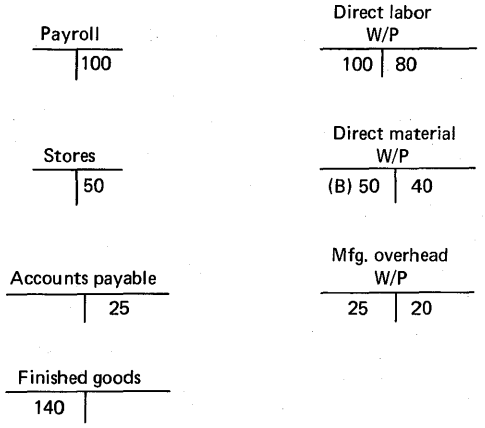


Figure 3.3 Posting for cost accounting.

Glossary of Cost Accounting Terms

Burden Same as *Overhead*

Cost accounting The maintenance of detailed records of expenditures incurred in producing goods or rendering services. All expenditures must ultimately be related to a finished product or service so that the total cost of each unit can be computed.

Direct labor Labor which directly affects or changes the nature of a product on which it is performed.

Direct material Any material which is used in the manufacture of a product and which forms an integral part of that product. It can easily be identified and its usage in the product can be readily determined.

Indirect labor Labor which is necessary to the operation of the factory, but which does not directly affect or change the product. Supervision, inspection, maintenance, and clerical labor are examples.

Indirect material Material used in the manufacture of a product but which is not an integral part of the product and is used in indefinite quantities.

Job order cost A system of cost accounting used in industries producing goods in production lots, each of whose specifications differs from those of the next lot.

Job record The medium for recording information pertaining to a particular job, operation, or process. It is contrasted with attendance record, which records the time an employee spends in the plant.

Job time Time an employee spends working on a particular job, operation, or process.

Overhead All expenses of factory operation not directly chargeable to a particular product. It includes indirect material, indirect labor, and other manufacturing expense.

Process cost A system of cost accounting used in industries producing similar type goods in a relatively continuous flow, such as steel strip, wire, and plate or sheet glass.

Standard cost Costs determined in advance of production and used as a measure of the efficiency of actual production.

Manufacturing Control

Manufacturing is fundamentally a process through which people and machines convert goods and materials from one form and value to another. The manufacturing process usually adds functional utility of some kind to materials or finished parts.

Nearly all manufacturing can be classified under one of the following general types of industries:

Basic production

Basic conversion

Fabrication

Standard assembly

Assembly with options

Assembly to specifications

Basic Production. This type of industry exploits natural resources to obtain raw materials for other manufacturers. A steel industry mines ore to produce pig iron which it then combines with other materials and scrap to produce steel ingots. Other examples are producers of aluminum, wood pulp, glass, petroleum, rubber, and mined or extracted chemicals.

Basic Conversion. The converter changes the products of the basic producer into a variety of industrial and consumer products. For example, the steel mill changes ingots into bars, tubing, rails, pipe, strip, wire, and structural shapes. Other converter products include paper, lumber, chemical derivatives, and plastics.

Fabrication. The raw products of the converter are transformed into a larger variety of products. Steel strip is stamped into bottle caps. Rods become nuts, bolts, and twist drills. Wire becomes nails and paper clips. Paper becomes bags, boxes, and newspapers. Glass is formed into bottles, light bulbs, and windshields.

Standard Assembly. In this process the products of the fabricators are made into finished goods of several or many parts. Well-known examples include radios, television sets, automobile engines, and many kinds of appliances.

Assembly with Options. Basic products and devices are assembled with a given range of options that can be chosen in various combinations by the customer. Examples include automobiles, airplanes, boats, and data processing equipment.

Assembly to Specifications. The manufacturer produces to the exact specifications of the customer. This category includes military hardware of all kinds, space exploration vehicles, ballistic missiles and defense systems, and atomic reactors.

Manufacturing control is a means of expressing the planned output of a manufacturing enterprise in terms of the required quantities and values of materials, facilities, and manpower. Means must be established to obtain sufficient information to control these functions.

The following are some of the main objectives of a manufacturing control system:

1. To provide the right quantities of materials and machines at the proper time. Future demands must be estimated and material planned for delivery to produce the anticipated amount of finished goods. Since most companies carry some inventory, what is on hand must also be related to what will be required. If it is not enough, additional material or parts must be ordered from suppliers. This may involve considerable *lead time* because the supplier may, in turn, have to produce components for delivery. To do this, he may have to allow lead time from *his* supplier, and so on.

2. To have ample manpower available with proper skills where needed. This includes both direct and indirect labor according to the job requirements of the plant.

3. To produce the products on schedule as called for by salesmen, distributors, or government agencies. This involves maximum utilization of machine time and facilities with manpower skills.

4. To produce all products in the most economical manner possible. Production management must have some system of record keeping in order to keep track of the progress of plant production and for an adequate system of cost accounting.

Management Functions

An effective manufacturing control system must be exceptionally well planned. From the planning of facilities to delivery of the finished product, at least the following management functions must be performed:

Forecasting
Materials planning
Inventory management
Scheduling
Dispatching
Operations evaluation

Forecasting. The forecast brings into consideration every known indicator concerning future market acceptance of the product. Essentially, the forecast attempts to predict the maximum quantity at the most profitable price. Also, some estimate is usually made about the length of time the product will be accepted by the consumer.

Materials Planning. Based upon the forecast, a plan for the procurement of materials must be produced. These may include raw or fabricated goods, or even partially assembled items for inclusion in a larger and more complex product.

Inventory Management. The proper management of inventories of both raw materials and finished goods is the means of carrying out materials planning.

Scheduling. A plan for facilities, equipment, and manpower must be integrated in such a way that the manufacturing cycle can begin and then continue according to design.

Dispatching. There must be a method of keeping the schedule functioning. Dispatching applies manpower to the schedule. It also provides a means for making minor adjustments to compensate for errors in the plan or changing conditions in its execution.

Operations Evaluation. This function evaluates execution of all plans on a continuing basis. It should signal in advance whenever revisions may be called for to meet the original objectives.

Glossary of Manufacturing Terms

Assembly Two or more parts or subassemblies put together as a unit to perform some specific function.

Engineering change A change in the material components or production methods of a part or assembly. The change may be made to improve performance or quality, to bring a product up to predetermined specifications, or to reduce the cost.

Machine tool load planning The functions involved in allocating the manufacturing load of a plant to fit the available machine tool capabilities.

Manufacturing or shop order The authority for the manufacturing floor to produce a given quantity of a part, subassembly, or assembly. The order also lists the operational steps required to produce the units; for example, cut, stamp, broach, mill, and polish. When each operation is completed, the order is posted and usually accompanies the job to the next operation. The completed order furnishes a record of the progress of the job through the plant.

Master bill of material List of the components or raw materials needed to manufacture a product or assembly.

Material or production explosion The quantity ordered times the number of component parts needed to manufacture one product or assembly.

Parts list List of all the parts required to make one product or assembly.

Piece or component part Raw material to which is added labor and burden to make one part.

Raw material Material in its basic state; for example, iron, wool, aluminum tubing, stainless steel sheets, various chemicals, cereal grains, etc. In many cases material used by one company to fabricate into other products is termed raw material. The *raw* material can be the finished product of another company. For example, bar stock steel is the finished product of a steel mill but raw material to a tool manufacturer.

Requirements planning Process by which a manufacturer determines the necessary quantities of product components (either parts or assemblies) so that they will be available in time for final assembly of the planned number of finished products.

Requirements, gross Total amount of material required to produce a stated quantity of finished goods.

Requirements, net Equal to gross requirements minus the amount on hand. Net requirements represents the additional amount of materials which must be obtained to produce a stated quantity of finished goods.

Sales forecast An estimate of future sales based on historical, current, and other pertinent data.

Subassembly Two or more parts or units which are incorporated into a subsequent assembly.

Where-used file A file in part number sequence with a reference to all products where each part is used.

Questions and Exercises

1. Why is it often necessary to rewrite orders received from customers or sales representatives? Can you think of any alternate to this requirement?
2. Name some of the accounting byproducts that can be obtained from a well-organized order writing procedure.
3. List the three most common types of information found on billing documents.
4. What are credit memorandums? Why are they used?
5. What accounting entries are made after an invoice is prepared?
6. List as many effects of a single sales transaction as you can.
7. Why are accounts receivable sometimes sold? Give examples of this type of business.
8. List the main steps of an accounts receivable procedure.
9. In a commercial enterprise, which accounting function is more important to the business, accounts receivable or accounts payable? Why?
10. Name some general types of industries. What function do they all have in common?
11. Name main management functions in a manufacturing environment. Explain why at least two of these functions are most important.

4 THE PUNCHED CARD AS A UNIT RECORD

The idea of punching holes in paper to record information was conceived for the textile industry nearly 250 years ago. In 1725 the Frenchman Bouchon devised a crude automatic control for draw looms by using a length of paper pierced with holes to represent a woven pattern. The paper was pressed by hand against a row of horizontal needles so that the needles opposite the blank spaces were pushed back while the remaining needles passed through the holes. This action was mechanically transferred to the positioning of the vertical or warp threads in the loom to form a pattern.

In 1728 an M. Falcon adapted the punched-hole idea to perforated cards, which he used to replace Bouchon's length of paper. To save space, Falcon arranged the needles or horizontal wires in multiple rows or ranks. He devised a square prism or "cylinder" for the cards where they could actuate the loom, one card at a time. Each card was placed in this "sensing" device by hand.

The next development was by an accomplished mechanic named Jacques de Vaucanson. He simplified the mechanism of the draw looms of that period and returned to the use of punched patterns on a continuous band of paper. Vaucanson also built an automatic feeding cylinder fitted with a rack wheel to pass one tooth for each change of the threads to automatically form the pattern. However, the operation of the wheel was quite complex and this device is not known to have been adopted.

In about 1790, a working mechanic of Lyons named Joseph Marie Jacquard invented a fishnet loom that won him an award from the London Society of Arts. While in Paris to receive the award, Jacquard was engaged by the Conservatorium of Arts and Sciences. Here, he later found the opportunity to make improvements in the weaving machine by studying the work of Bouchon, Falcon, and Vaucanson. Apparently he combined the best qualities of his predecessors to produce the industrial loom that still bears his name (Figure 4.1). The machine was so successful that by 1812 there were an estimated eleven thousand looms in France. The loom was declared public property in 1806; as a reward for his accomplishment, M. Jacquard was given a pension and a royalty on every machine. Four years after his death a statue was erected in his honor in Lyons.

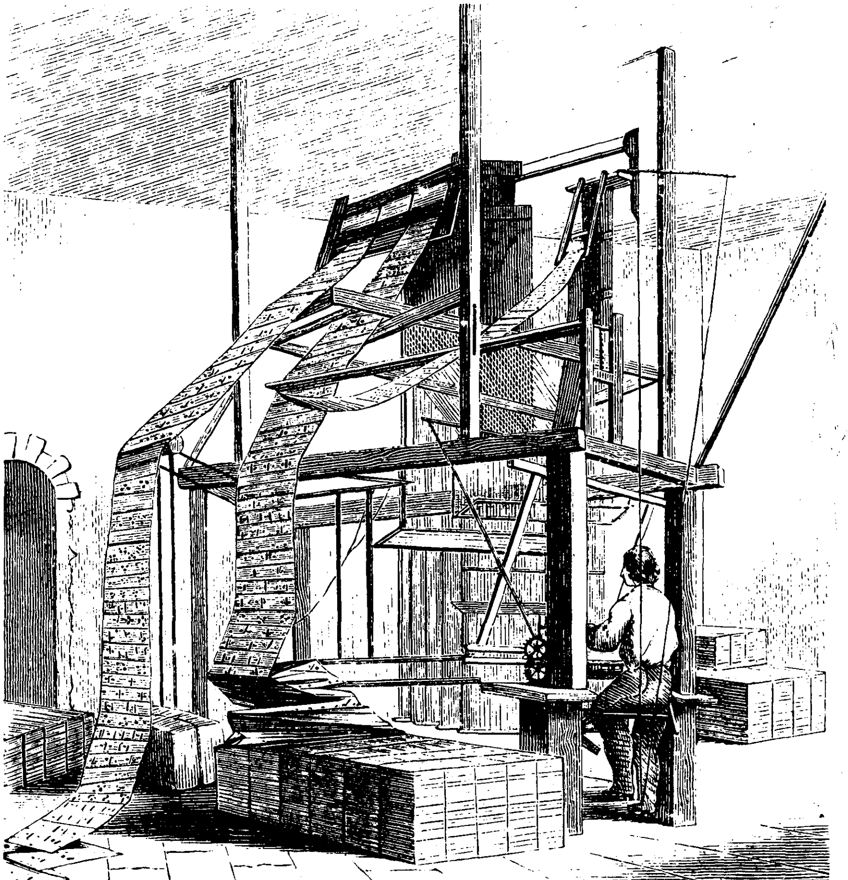


Figure 4.1 Jacquard loom.

To set up the machine for weaving, a pattern was first laid out on ruled graph paper to represent the designer's original *weave draft.* The draft showed at least one repeat or complete weave unit of the fabric to be produced. From the drawing, one card was punched out to represent each throw of the shuttle carrying the horizontal or *weft* thread through the design. Machines were also invented to mechanically punch and duplicate the cards.

To operate the loom, cards were laced together with loops of string to form a continuous chain. As the cards moved into place on the cylinder (actually a quadrangular block), some needles passed through the punched holes and thereby raised the corresponding warp threads. Where there were no holes, needles were pushed back by spring action and the corresponding warp threads were not raised. Any woven pattern could be repeated indefinitely simply by using the same cards over and over. By adding several Jacquard attachments to one loom, a weaver could produce not only intricately figured fabrics but also pictures of considerable size. One of the most elaborate patterns woven at that time was a famous portrait of M. Jacquard himself. It required 24,000 punched cards.

This unique use of information stored as punched holes was undoubtedly one of the earliest examples of an artificial memory. That is, information was stored in such a way that it could be made directly accessible to a machine without human intervention of any kind. This important concept is a basic function of both the design and operation of modern data processing equipment.

Data Representation for Machines

The task of efficient data management has been a perennial business problem since the beginning of organized commerce and industry. In chapter 3 it was shown how very large quantities of data can originate in almost any modern business environment. It is hardly necessary to point out that the same problem is also acute in every government, scientific, and military endeavor. The "paper blizzard" is everywhere.

The best way to cope with this situation is to process data with machines. Since the majority of products and services reaching the market today are produced by more or less automated methods, it would seem to follow that the processing of information should also be carried out more efficiently by machine. However, in considering this solution to the problem, it is also readily apparent that most data does not originate in a form that is suitable for mechanical or electronic manipulation. Therefore, the first step in automated processing is to record data in such a way as to make machine processing practical. There are two basic requirements for any system of data representation.

First: The method used must be one of *preestablished convention.* More simply, there must be some sort of specific agreement about how such representation is to be formed, used, and understood by machines or people or both. Data and information are usually represented for people as symbols (Figure 4.2).

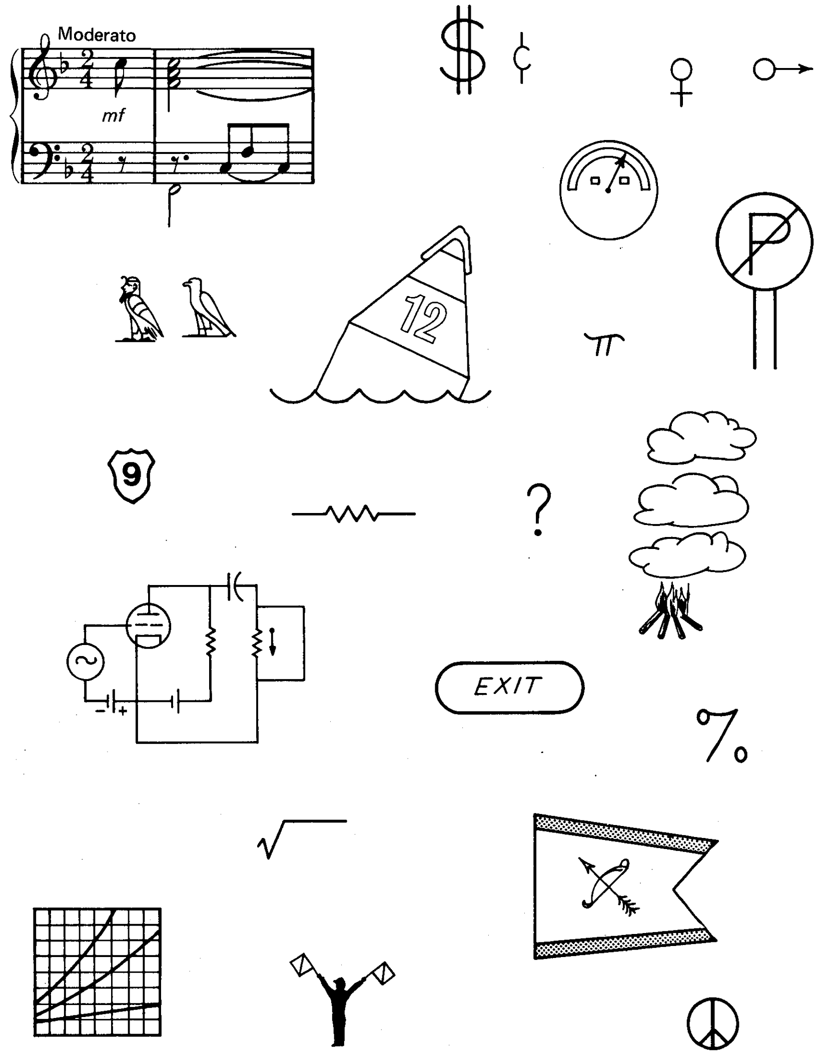


Figure 4.2 Data represented by symbols.

For example, the printed characters on this page are really symbols. When understood, they convey meaning to the reader in a manner established by education, training, and experience. A symbol may convey one meaning to some people, a different meaning to others, and no meaning at all to those who do not understand its significance. But written symbol representation is not generally suitable for machine use (although specialized machines have been developed recently to read both printed and handwritten symbols directly from paper documents).

Second: The method of representation used must be *characteristic of the vehicle of communication.* For example, when data are transmitted over wire, the message is represented as a pattern of precisely generated electrical impulses. That is, the data are coded suitably for this type of transmission in a form that can be sent and received by machines. Data are represented by some instruments as graphs on ruled paper. Others display data on gauges, dials, or cathode ray tubes.

We are told that some primitive peoples developed the ability to talk to each other over long distances by beating on drums or forming smoke signals. These conventions were devised for rather unusual vehicles of communication.

It is therefore a general rule that whenever data are to be processed by a machine, the data must first be reduced to some exactly specified set of symbols, codes, or patterns which can be readily sensed and interpreted by a machine. These symbols or patterns usually differ from those commonly used by people, because convention must conform to the design and operation of the machine. The choice of data representation and the way the data will be interpreted is a matter of convention on the part of engineers, designers, and manufacturers.

Dr. Herman Hollerith is reported to have conceived the idea of recording U.S. census data in cards while traveling by train. He noticed the conductor punching holes in the railway tickets in a certain pattern to identify each passenger and destination. The holes represented specific items of information which the conductor could interpret according to the rules or conventions he had established.

Dr. Hollerith recognized yet another important aspect of machine data processing. There must be a standard *medium* in which to record information, a medium of practical size and form made from some material of known characteristics. The early French inventors of the automatic loom had called for the "best obtain-able paper with a weight of about 16 or 18 cards to the pound."

As machines become more and more advanced in data handling capability, other recording media have gained wide acceptance. Punched paper tape and magnetic plastic tape are commonly used by computers to read and write data at extremely high speeds. Many computers use still more advanced recording media in the form of magnetic surfaces on rapidly revolving metal disks and drums.

The 80-Column Card

Modern punched-card documents are still often called *Hollerith cards*, after their inventor. It is also reported that when Dr. Hollerith was asked what size card should be used in his new machines, he casually chose the dimensions of a U.S. Treasury note he happened to have in his pocket at the time. Whether or not this story is true, the standard 80-column card is the exact size of the old United States paper money: $7\frac{3}{8}$ inches long by $3\frac{1}{4}$ inches wide.

More frequently, the name *IBM card* is used, referring to the International Business Machines Corporation, the largest manufacturer of card-operated equipment. When first used commercially, IBM cards were punched with round holes. In 1928, the card capacity was increased from 45 to 80 *columns* by using smaller, rectangular holes. IBM had concentrated on the development and marketing of electromechanical machines. The rectangular shape provided better electrical contact through the holes as the cards were fed under metal sensing brushes.

It is always a minor point of semantics whether *punch* card or *punched* card is the proper term. Either may be used. *Punch* card is used in the general sense to relate the blank document to the punching machine which produces the holes. *Punched* card designates a card after it has been perforated with holes ready for some sort of machine processing. Card machine operators sometimes refer to their documents as *tabulating* cards, after the *tabulator*, the first machine to produce printed reports from punched cards.

All card paper stock is manufactured to rigid specifications to guard against impurities and to insure uniform characteristics under operating conditions. The stock is 0.007 inch thick and must be electrically nonconducting.

The actual punching area is divided across the length of the card into 80 equal spaces or vertical *columns*, numbered from left to right (Figure 4.3). Each column is subdivided into 12 punching positions: 0 through 9, 11, and 12. The 11 and 12 positions are at the top of each column, above the 0 through 9 positions. These two positions are sometimes referred to as the X and R positions.

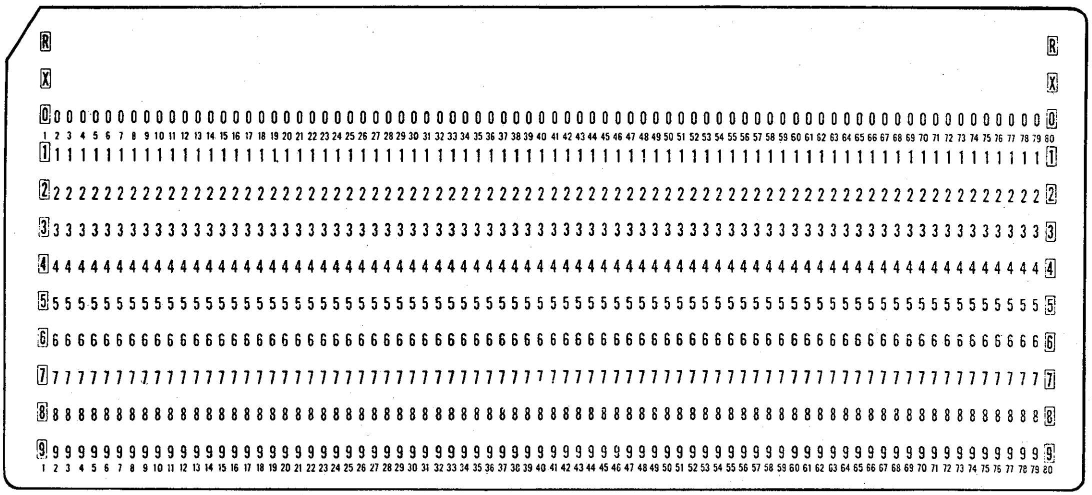


Figure 4.3 80-column card.

The 12 punching positions form equally spaced horizontal *rows.* The bottom row contains all the 9 positions, the next row above all the 8 positions, the next all 7s, and so on. The rectangular holes are always punched out precisely at the intersections of rows and columns. Therefore, a single card document has as many as 80 X 12, or 960 possible hole positions. However, in actual use the card is never completely punched out.

A punching position can be assigned almost any predetermined significance, provided the chosen recording convention can be recognized by the equipment that is to process the cards. When a hole is punched, it means that the value, data, or other significance is recorded. When a punching position is left blank, it means the information is not recorded. Most important, the punched hole is permanent. Recorded information cannot be changed or erased. Of course all recording is restricted to the 960-hole capacity of the card.

Punched cards are *unit* records because a card is always punched, read, or otherwise processed as a single unit of information. However, if capacity permits, more than one record can be punched in a single card.

A group of cards, usually forming a given set of data, is called a *deck.* After cards have been placed in some orderly sequence for processing, the deck becomes a *file*: the payroll file, the accounts receivable file, the address file, etc.

Because of their exact uniformity, a means must be provided to keep all cards in a deck or file facing in one direction, rightside-up. This is particularly important for machine handling. For this purpose, cards are made with specified *corner cuts.* Any card out of position can be detected immediately when a corner cut does not match other properly aligned cards in the file, as shown in Figure 4.4.

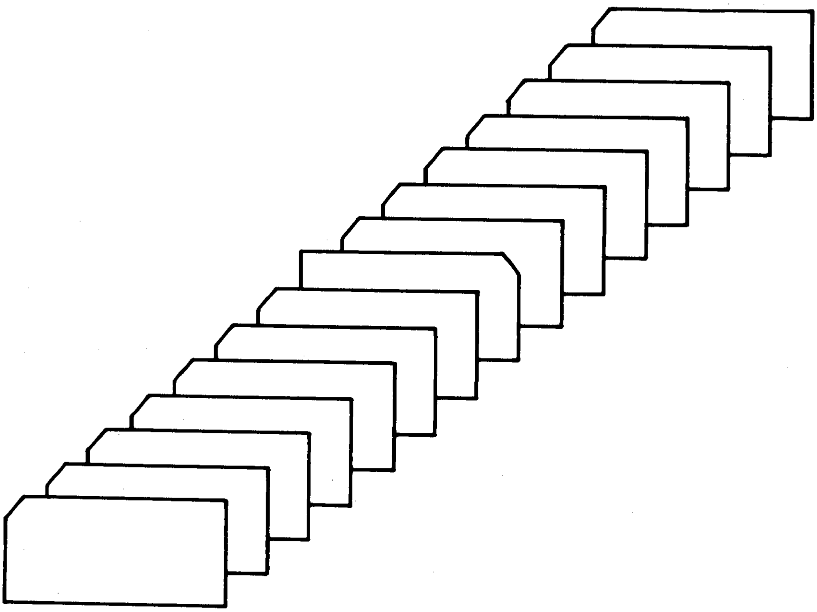


Figure 4.4 Opposing corner cuts in a card file.

Cards are also available from the manufacturers in solid colors or with distinctive colored stripes. The colors are useful in identifying different types of cards in the same file. For example, in a name and address file, the name cards may be salmon with an upper right corner cut; the address cards may be plain manila with an upper left corner cut. In this case the corner cuts and colors serve to identify each separate name and address item in a file. Colors are used only for clerical and manual manipulation of cards; colors are not recognized by machines.

IBM Card Code

The standard IBM card code uses the twelve possible punching positions in a vertical column to represent the digits 0 through 9, the letters of the alphabet, and the special characters of punctuation and report printing. The digits 0 through 9 are represented by single holes in the numeric rows. For example, to represent the digit 1, a hole is punched in the one row. To represent the digit 5, a hole is punched in the five row, but not in the same column with the 1 just punched. To represent 0, a hole is punched in the zero row, also in a different column. Therefore, to record the quantity 150, three holes are punched in adjacent columns: 1, 5, and 0. Refer to Figure 4.5. The choice of columns is a matter of card design and convenience, a subject that will be discussed later.

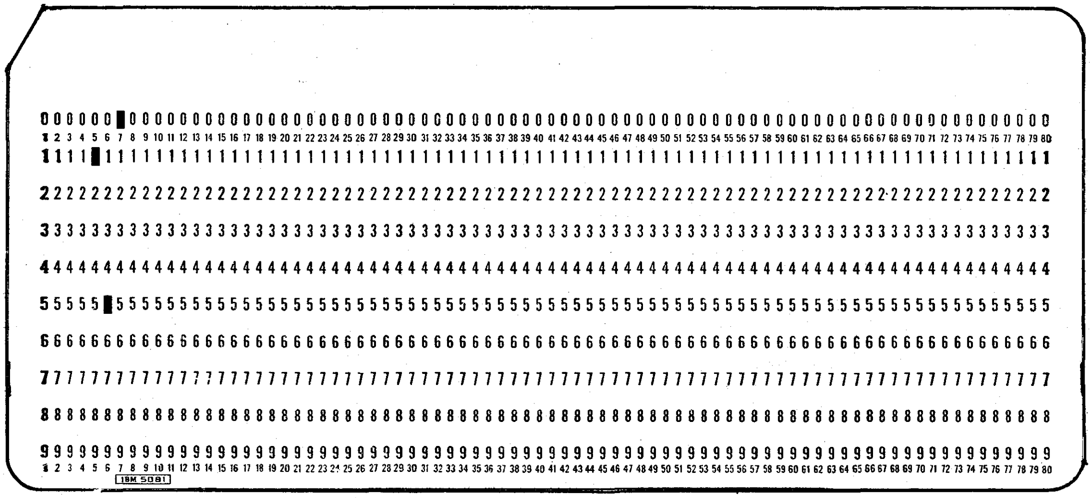


Figure 4.5 Digits 1–5–0 punched in adjacent card columns.

To represent the letters of the alphabet, the card is thought of as being divided into two sections—the *numeric* section just explained, and a *zone* section. The zone section is made up of the zero, eleven, and twelve rows. Note that the zero row is common to both the numeric and zone sections of the card. To represent the letter A, two holes are punched in a single column: 12 and 1. To represent B, two holes are also punched: 12 and 2. The letter C is 12 and 3; D is 12 and 4; and so on through the letter I, which is 12 and 9.

The eleven row is used in combination with the numeric hole positions to represent the letters J through R; holes 11 and 1 represent J, holes 11 and 2 represent K, and so on. The zero row in combination with numeric positions is used to represent the letters

S through Z. Since there are 26 letters of the alphabet and 27 possible zone and numeric combinations, the combination of holes 0 and 1 is omitted in the coding scheme. Holes 0 and 2 represent the letter S; holes 0 and 9 represent Z.

To record alphabetic information, adjacent card columns are normally chosen, depending upon the format of the record. To record the name "Jones," for example, five card columns are required, one for each letter in the name. Refer to Figure 4.6.

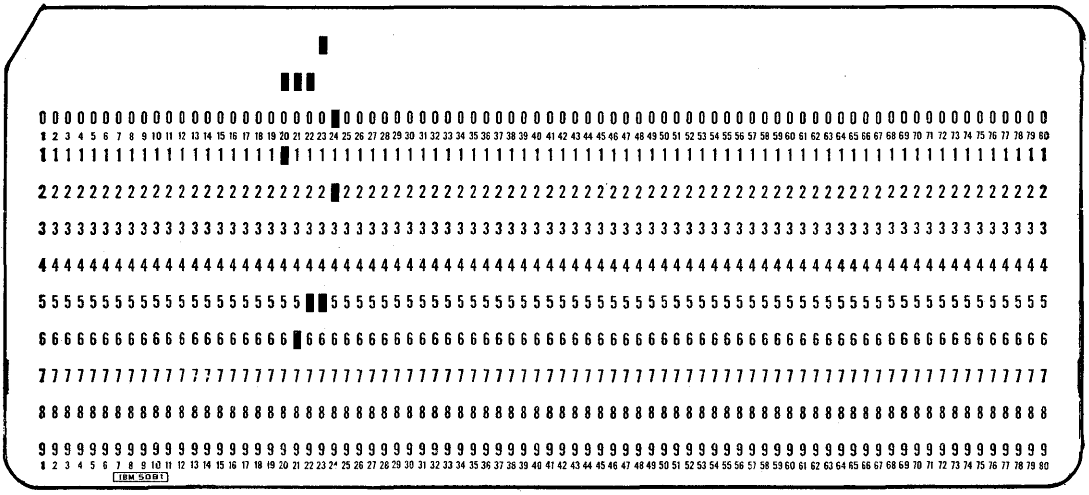


Figure 4.6 Letters J–O–N–E–S punched in adjacent card columns.

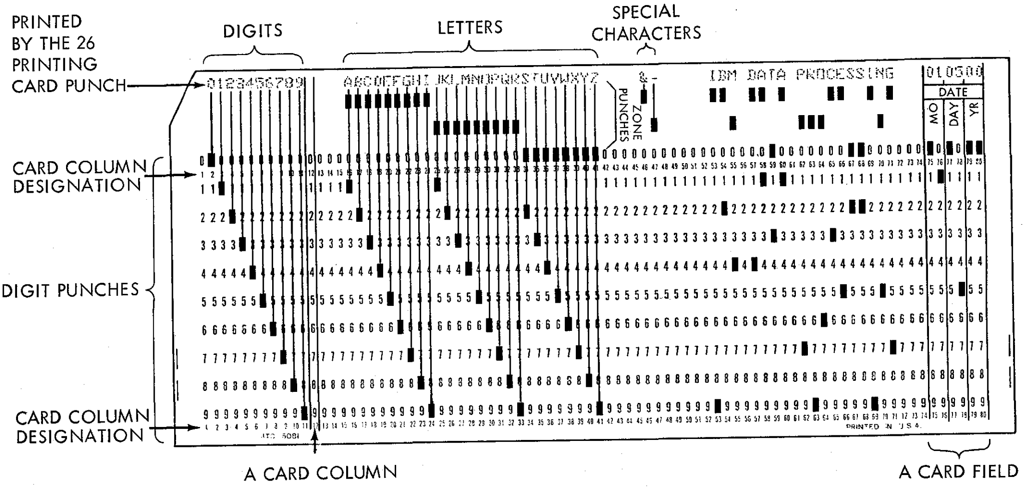


Figure 4.7 80-column card code.

Figure 4.7 shows the complete IBM card code as just described, plus two additional zone punches to represent the ampersand (&) and minus or dash (—) characters. The printing at the top of the card is done by a machine called a printing keypunch, a key-operated device that prints the information corresponding to the holes in each column.

Standard BCD Interchange Code

The previously described IBM card code is accepted as the standard punched hole representation of alphabetic and numeric data when used with IBM unit-record equipment. However, punched cards also serve as an important data recording medium for many computer systems. To provide compatibility of data for interchange among these various systems, the IBM card code has been extended to represent a total of 64 different characters. Some of these characters are useful in the representation of mathematical formulas and in message transmission over long-distance communication equipment. Other characters mark the end of records, fields, and blocks of information recorded on magnetic and paper tape. Some characters, such as the familiar $, #, *, etc., are used when printing reports, checks, invoices, statements, and the like. A list of these special characters with their proper names is shown in Figure 4.8.

The letters BCD stand for *binary coded decimal*, a method of coding information. It is related to the binary method of arithmetic which is widely used in the internal computation and data handling units of computers. Figure 4.9 shows the complete BCD code in *collating sequence*; that is, the sequence in which computers can be programmed to sort data. Note that the first column of the illustration shows the standard characters or graphics that are coded. Five of the standard BCD combinations print out as either of two characters, depending upon the type set used in the printing device. The two variations are called graphic subset 1 and graphic subset 2 (Figure 4.10).

Graphic subset 1 is used primarily for computer report writing and most commercial applications. Graphic subset 2 is used when writing or printing advanced programming languages for computers as it meets general requirements for mathematical symbolism.

The second column of Figure 4.9 shows the punched hole combinations for each graphic. The third column shows the *bit* code representation for internal computer storage. This column is

SYMBOL	NAME
⧧	Group Mark
‡	Record Mark
⧻	Segment Mark
⁓	Word Separator
@	At Sign
#	Number Sign
&	Ampersand
+	Plus
*	Asterisk
%	Percent
/	Slash
\	Backslash
□	Lozenge
b	Blank
b̸	Substitute Blank
(	Left Parenthesis
)	Right Parenthesis
[	Left Bracket
]	Right Bracket
√	Tape Mark
<	Less than
>	Greather than
=	Equal to
;	Semicolon
:	Colon
•	Period or Point
'	Prime or Apostrophe
—	Minus or Hyphen (Dash)
Δ	Delta

Figure 4.8 Special characters or symbols with names.

BCD Code	Graphic Subset 1 Print Arrangement A	Graphic Subset 2 Print Arrangement H
8-2-1	#	=
8-4	@	'
A-8-4	%	(
B-A	&	+
B-A-8-4	□	)

Figure 4.10 Graphic subsets 1 and 2.

COLLATING SEQUENCE

	CHARACTER Report	CHARACTER Program	CARD CODE	BCD CODE (Core Storage)						
low →	b		No Punches	C						
	•		12-3-8		B	A	8		2	1
	□	)	12-4-8	C	B	A	8	4		
	[		12-5-8		B	A	8	4		1
	<		12-6-8		B	A	8	4	2	
	⧧		12-7-8	C	B	A	8	4	2	1
	&	+	12	C	B	A				
	$		11-3-8	C	B		8		2	1
	*		11-4-8		B		8	4		
	]		11-5-8	C	B		8	4		1
	;		11-6-8	C	B		8	4	2	
	Δ		11-7-8		B		8	4	2	1
	—		11		B					
	/		0-1	C		A				1
	,		0-3-8	C		A	8		2	1
	%	(	0-4-8			A	8	4		
	⁓		0-5-8	C		A	8	4		1
	\		0-6-8	C		A	8	4	2	
	⧻		0-7-8			A	8	4	2	1
	b̸		2-8			A				
	#	=	3-8				8		2	1
	@	'	4-8	C			8	4		
	:		5-8				8	4		1
	>		6-8				8	4	2	
	√		7-8	C			8	4	2	1
	?		12-0	C	B	A	8		2	
	A		12-1		B	A				1
	B		12-2		B	A			2	
	C		12-3	C	B	A			2	1
	D		12-4		B	A		4		
	E		12-5	C	B	A		4		1
	F		12-6	C	B	A		4	2	
	G		12-7		B	A		4	2	1
	H		12-8		B	A	8			
	I		12-9	C	B	A	8			1
	!		11-0		B		8		2	
	J		11-1	C	B					1
	K		11-2	C	B				2	
	L		11-3		B				2	1
	M		11-4	C	B			4		
	N		11-5		B			4		1
	O		11-6		B			4	2	
	P		11-7	C	B			4	2	1
	Q		11-8	C	B		8			
	R		11-9		B		8			1
	‡		0-2-8			A	8		2	
	S		0-2	C		A			2	
	T		0-3			A			2	1
	U		0-4	C		A		4		
	V		0-5			A		4		1
	W		0-6			A		4	2	
	X		0-7	C		A		4	2	1
	Y		0-8	C		A	8			
	Z		0-9			A	8			1
	Ø		0	C			8		2	
	1		1							1
	2		2						2	
	3		3	C					2	1
	4		4					4		
	5		5	C				4		1
	6		6	C				4	2	
	7		7					4	2	1
	8		8				8			
high →	9		9	C			8			1

Figure 4.9 Standard BCD code in collating sequence.

for reference only and does not pertain to any card representation of data.

Notice again that the assignment of a particular meaning (in this case, a character) to a punched hole is a choice that can be made by the user. Printers can be obtained with typesets specified for almost any unique application of punched cards. For example, machines used in the United Kingdom may have little or no occasion to print dollar signs. Instead, equipment is required to print the signs for pounds, shillings, and pence. Other countries and locations may also have individual requirements for typesets to print peculiar language and monetary graphics. The coding system just described can be used to represent an almost unlimited variety of data.

Card Fields

All records are made up of a number of related items that may be called *fields*. Figure 4.7 also shows a card field at the right end of the card, made up of columns 75 through 80 and labeled Date. In this field, columns 75 and 76 are reserved for the recording of month, expressed in numeric fashion. That is, January is punched as 01, February as 02, and so on. Columns 76 and 77 are set aside to record the day of the month, also expressed numerically. Columns 79 and 80 record the last two digits of the year. The design of the field illustrates a number of considerations that must be taken into account whenever data is punched into cards.

The total amount of information to be put into the card is determined by the requirements of the reports and other documents that may be prepared from it later. A card field may be from one to 80 columns in length, depending upon the length of the particular item of information to be punched. For example, a field reserved for name would usually be longer than a field reserved for social security number. However, each field must be of consistent length in all related records making up a file. Also, the field must be positioned in the same columns in all cards of the file. And, once a field is assigned to record some specific item of information, it cannot be used to record any other type of data.

This restriction in card layout and design is true of hand written documents as well. Although the number of digits or characters may not be set exactly, the location of most items on any form is indicated. For example, one of the standardized forms in common use is the personal check. Each item of information on this record is located in a designated position.

The length of a record field must be known exactly for any type of machine recording and processing. The length is determined by the maximum number of digits and characters that can be expected to be recorded in the field. For example, social security number is always nine digits in length. Therefore, nine columns are always required to record that item of data in a card record.

To punch any quantity or value, the number of columns reserved must be sufficient to record the largest quantity expected in any record. Suppose the amount of sale is one of the items to be punched in a department store accounts receivable application. If the selling price of no item in the store is as much as $10,000.00 but might be more than $1,000.00 then, six columns is all that will ever be required to record this amount. (Dollar signs, and decimal points are not recorded directly into the amount field. The postion of these characters is understood in the format of the record and its component fields.)

From right to left, the positions in a numeric field are known as the units positions, tens position, hundreds position, thousands position, and so on. Each digit of any number recorded in a card field must be punched into its corresponding column. Therefore, if the number to be punched is shorter than the number of columns reserved for that field, *insignificant zeros* must be added at the *left* of the number to fill the field.

For example, if the number to be punched is 764 and the card field is five columns long, the number is punched as 00764. If the number is 64, the field is punched as 00064. The single 4 is punched as 00004. As previously explained, it follows that fields of any machine records must be laid out in advance to accommodate the maximum number of characters expected. Data cannot exceed field size.

To record an alphabetic item such as name, punching starts properly in the leftmost column of the field and continues on to the last letter to be punched. If the alphabetic item does not fill the field, the remaining columns of that field are skipped over, unpunched. Thus the constant length of the field is always maintained.

Punched cards are defined as *fixed-length* records made up of *fixed-length* fields. Some types of computers can process and manipulate records of variable length, particularly where records are magnetically recorded on tape. In this case, the amount of data per record is not limited by the physical characteristics of the recording medium.

The 96-Column Card

In July, 1969, IBM announced the most radical change in the physical characteristics of the punched card since its introduction by Herman Hollerith in 1890. As the principal recording medium for the System/3, a small computer, the new card is slightly more than one-third the size of the old 80-column card: $3\frac{1}{4}$ inches long by $2\frac{5}{8}$ inches wide. By recording with small round holes, 96 columns of information can be punched in a single card. Thus, the new card holds 20% more data in about 30% of the old card area. Figure 4.11 shows the layout of the IBM 96-column card.

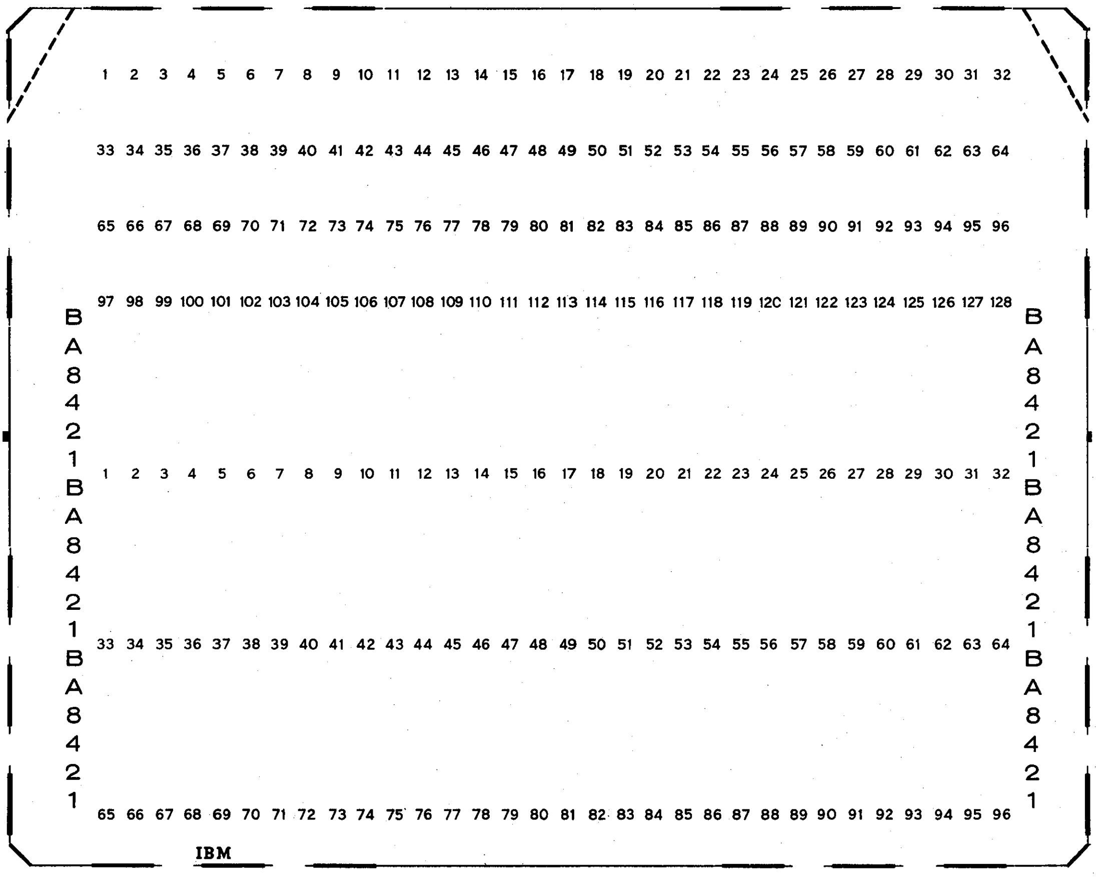


Figure 4.11 96-column card.

The card is divided into two main recording areas: print and punch. The print area is subdivided into four horizontal rows of 32 print spaces each. The spaces are numbered from left to right;

the top row from 1 to 32, the second row from 33 to 64, the third row from 65 to 96. Print spaces correspond to the numbered punch columns in the lower section of the card. Print line 4 does not have a corresponding tier but may be used for printing. The printing can be done at the time the card is originally punched, either by a key-operated data recorder or during System/3 processing.

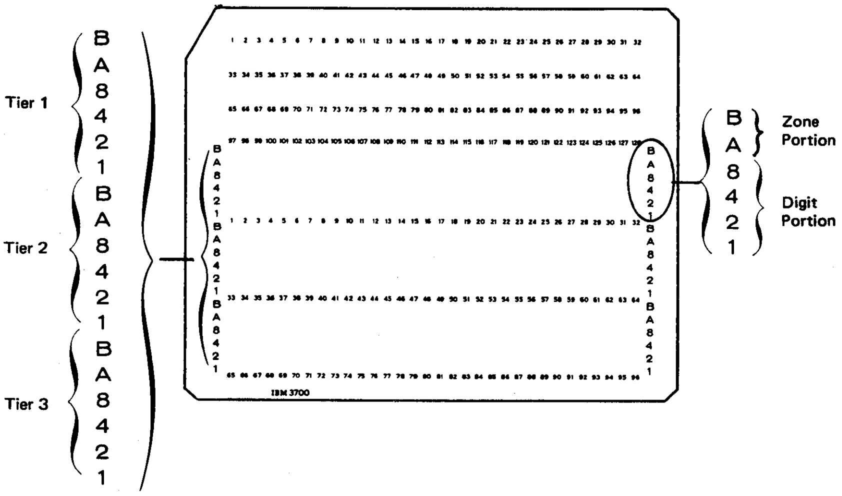


Figure 4.12 96-column card punching area divided into tiers.

The punch area is divided into three horizontal *tiers*, each with 32 vertical columns (Figure 4.12). However, whereas the 80-column card has 12 punching positions in a single column, the 96-column card has only six. In these positions, all digits 0 through 9, the letters of the alphabet, and 28 special characters can be represented.

The numeric portion of each column has four punching positions: 1, 2, 4, and 8 (Figure 4.13). The zone portion of each column has two positions for punching, called A and B. Like the 80-column coding scheme, digits are represented by punched holes in the numeric portion of a column. But, *unlike* the 80-column code, the four numeric positions are assigned a *value* in the code. Thus, the digit 1 is represented as a hole in the one position; 2 is a hole in the two position, while 3 is represented by two holes in the same column with the value of 2 *and* 1. The digit 4 is a hole in the

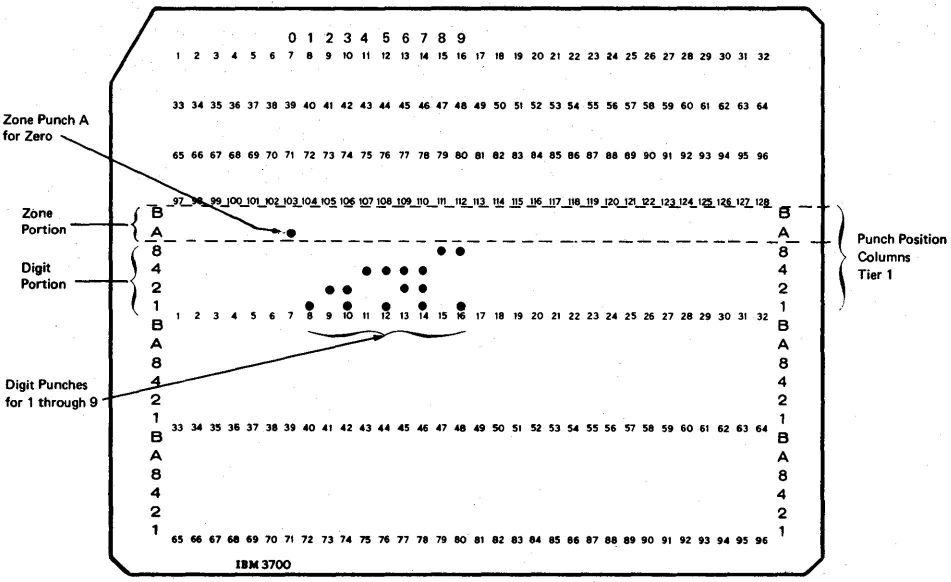


Figure 4.13 96-column card. Numerical punching positions.

four position; 5 is a hole in the four and one positions in the same column. The digit 6 is punched four and one; 7 is four, two, and one; and so on. Because there is no zero position, the 0 is repre-

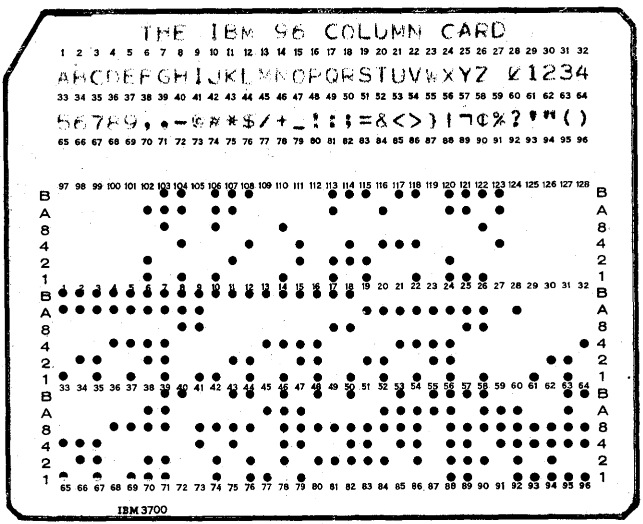


Figure 4.14 96-column card. Numerical, alphabetical punching and printing.

sented by a single hole in the A zone position. By referring to Figure 4.9, it can be seen that numeric punching follows the BCD Interchange Code scheme, with the exception of the code for 0.

			Numeric Characters									
			0	1	2	3	4	5	6	7	8	9
Punch Positions	Zone	B										
		A	A									
	Digit	8									8	8
		4					4	4	4	4		
		2			2	2			2	2		
		1		1		1		1		1		1

			Alphabetic Characters																									
			A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Punch Positions	Zone	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B								
		A	A	A	A	A	A	A	A	A	A										A	A	A	A	A	A	A	A
	Digit	8								8	8								8	8							8	8
		4				4	4	4	4						4	4	4	4					4	4	4	4		
		2		2	2			2	2				2	2			2	2			2	2			2	2		
		1	1		1		1		1		1	1		1		1		1		1		1		1		1		1

			Special Characters																											
			}	¢	.	<	(	+	|	!	$	*	)	;	¬	-	/	&	,	%	—	>	?	:	#	@	'	=	"	b̸
Punch Positions	Zone	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B														
		A	A	A	A	A	A	A	A								A	A	A	A	A	A	A							
	Digit	8		8	8	8	8	8	8	8	8	8	8	8	8			8	8	8	8	8	8	8	8	8	8	8	8	
		4				4	4	4	4			4	4	4	4					4	4	4	4			4	4	4	4	
		2		2	2			2	2	2	2			2	2			2	2			2	2	2	2			2	2	
		1			1		1		1		1		1		1		1		1		1		1		1		1		1	

Figure 4.15 Binary coded decimal representation of 96-column card characters.

Alphabetic characters are represented by a combination of punches made in the numeric (8, 4, 2, 1) and zone (A, B) portions of a column, as shown in Figure 4.14. Special characters are represented by a combination of punches in the digit and zone portions of the column. Figure 4.15 shows the combinations used to represent each character.

Questions and Exercises

1. Explain two requirements for representing data for machines.
2. Describe the effective punching areas of the 80-column card; the 96-column card.
3. List some different types of recording media with which you are familiar.
4. What is probably the most important advantage of punched-hole recording?
5. Explain the use of punched card corner cuts.
6. What is a collating sequence?
7. Explain why fields of machine records must be explicitly designed.
8. What are some of the advantages and disadvantages of the 96-column card?
9. What is the BCD code?
10. What is the purpose of printing on punched cards?

THE TRANSCRIPTION TO PUNCHED HOLES 5

Data processing is a series of planned operations and actions upon data to achieve some desired result. The procedures and devices used make up the *data processing system*. The devices may vary widely. They may be complex machines or only paper and pencil. The procedures, however, always remain basically the same.

Of course, there are many types of data processing systems. These systems vary in size, complexity, speed, cost, and application. But whatever is to be processed or whatever equipment is to be used, at least three fundamental considerations are involved:

1. The source data, or *input* entering the system
2. The orderly, planned *processing* within the system
3. The end result or *output* from the system

In some computer systems, data can be handled on a *real time* basis. That is, information in some form enters the system *as it happens*. For example, a pressure gauge in an oil refinery transmits a reading directly to an input device capable of translating that reading into input data the system can operate upon. After some processing, an output device directly actuates a control to close a valve and regulate pressure. Other readings of volume and temperature also may be fed to the computer as changes occur or at periodic intervals. Data processing is programmed in such a way that output from the computer directly controls the production of the refinery.

In a punched-card system, input is always *historical.* That is, all data entering the system has originated some time previously. This chapter describes the various methods of transcribing this sort of data into punched cards.

80-Column Card Punching

Data to be recorded on punched cards are very often manually transcribed from some original source document. This process, known as *keypunching*, is done by some type of card punching machine especially designed for this purpose. Keypunches are equipped with a keyboard similar to the one on a typewriter. An operator depresses keys to punch holes just as a typist depresses keys to print letters on paper. When each key is depressed, the character code corresponding to that key is punched as one or more holes in a single column of a card. As the operator reads from an original document, the information is copied exactly in the form of punched holes, just as a typist might copy from a handwritten document to produce a typewritten page.

Many different models of keypunches can be obtained. Typically, 80-column cards pass through the punch one at a time from a hopper to a punching station. After punching, each card is automatically released to a reading station. The card then passes through this station in phase, column by column, with the following card now at the punching station. Whenever a card is completely punched at the punching station, a card at the reading station moves into the stacker.

Because each card passes the reading station in phase with the following card at the punching station, it is possible to have automatic duplication from card to card. That is, data from a specific column at the reading station may be read and transmitted back by the machine to be punched in the corresponding column of the card following at the punching station. When this card then reaches the reading station, the same character is again transmitted back to the next card. This process continues as long as cards are fed through the machine or until the *duplicating* is otherwise controlled by the operator.

The ability to duplicate or to reproduce data automatically is an important function of card punching in particular and of all data processing systems in general. Manual punching and keying may be considerably reduced, thereby decreasing the chance of human error in the processing. Data common to any group or set of records, such as date originated, may be established for the first card or record and automatically duplicated in all succeeding records in the deck or file.

Figure 5.1 shows an IBM 129 Card Data Recorder. The basic components of the machine are labeled in the photograph.

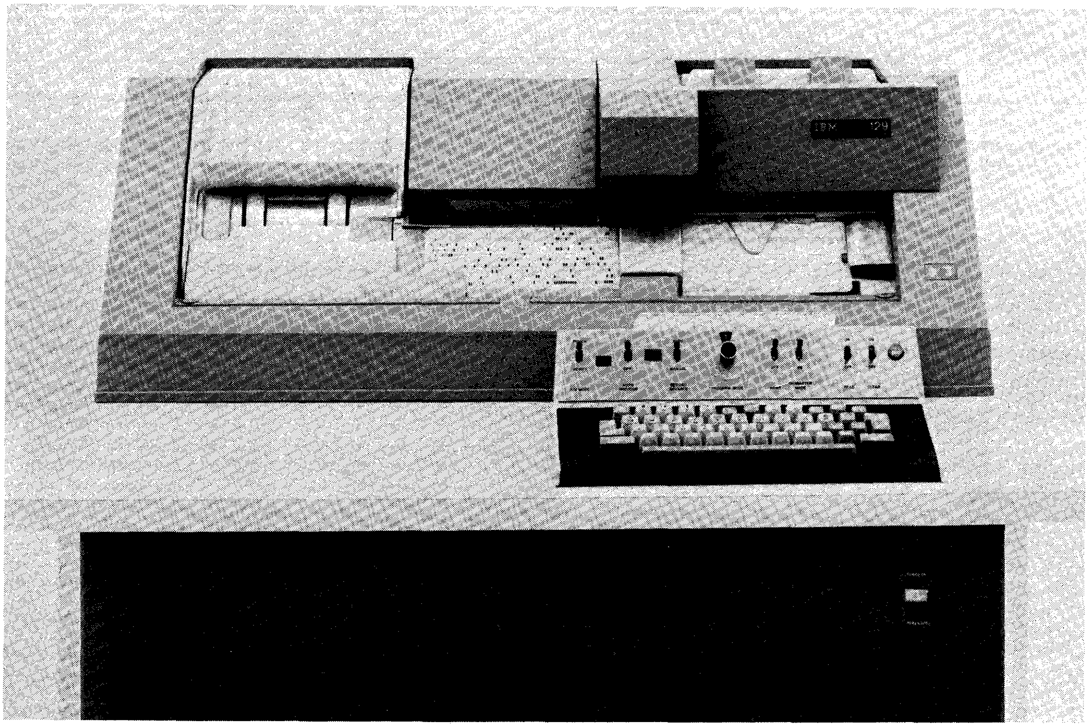


Figure 5.1 IBM 129 Card Data Recorder.

Card Hopper

Blank cards to be punched are placed in the card hopper, a receptacle located in the upper right section of the machine. A sliding pressure plate at the back of the hopper pushes the cards forward against a set of feeding knives. When the feed key on the keyboard is pressed, the knives are actuated to move the front card down into a feed station. The card is held at the feed station for registration at the punching station. The hopper holds about 500 cards, printed or *face* side forward, 9-edge down.

The first two cards from the deck are fed by pressing the feed key. All following cards can be fed automatically by setting an auto feed switch. Operator control switches are just above the keyboard as shown in Figure 5.2.

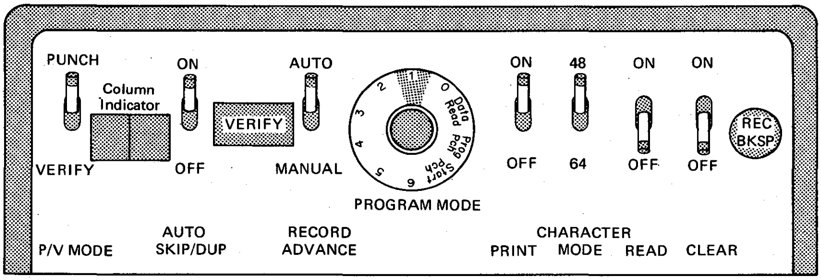


Figure 5.2 129 Card Data Recorder. Operator control switches.

Punching Station Punching is done at the second station along the card path. As stated above, when normal punching is begun, two cards are fed into the card bed at the right of the punching station. As the second card is fed, the first card is registered for punching. That is, the left end of the card is accurately positioned under the punch dies.

While the first card is being punched, the second card is held at the feed station to the right of the first card. When the first card is completely punched, it is moved either automatically or under key control of the operator to the reading station. At the same time, the second card is registered at the punching station while the *third* card is moved from the hopper to the feed station.

The second card is punched. When completed, it also is moved to the reading station while the first card is ejected to the stacker in the upper left section of the machine. Punching continues, card by card, in series. As each card is fed from the hopper to the feed station, a card is also ejected to the stacker. Thus, the operator controls a continuous flow of cards through the machine from right to left. Provision is made for single cards to be inserted by hand if necessary.

Stacker Cards are ejected by metal fingers that grasp the upper or 12-edge of the card. The fingers are located on a roller mechanism which stacks the cards horizontally, building the deck by adding cards, one at a time, to the bottom of the stack. Note that this arrangement keeps the cards in the sequence in which they are punched; first card on top, last card on the bottom of the deck, 12-edge to the rear. As will be shown later, it is important in most applications to preserve the exact punching sequence for a machine verification of the cards in another keying operation.

A card weight holds the stacked cards, in position until removed by the operator.

Keyboard

The keyboard of the card punch is somewhat like a typewriter keyboard. Figure 5.3 is a typical key arrangement. The keyboard can be rotated on the desk or reading table for operator convenience and comfort.

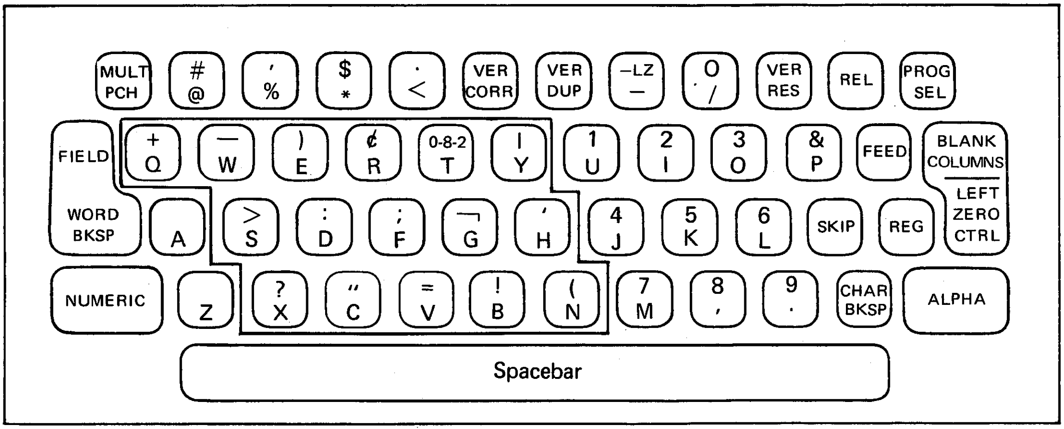


Figure 5.3 IBM card punch key arrangement.

Letters of the alphabet are placed in the same locations as on the standard typewriter. However, whereas the numeric row of keys is placed at the top of a typewriter keyboard, these keys are superimposed upon the alphabetic keys of the card punch. For example, the digit 1 is on the same key as the letter U, the digit 2 is placed with the letter I, 3 is with O, and so on. The effect is to arrange the digits in a convenient grouping while at the same time maintaining the standard typewriter touch system. This permits the operator to punch an alphabetic field with both hands and then, without moving fingers from the *home* position, to punch a numeric field with the right hand only. Use of the right hand alone frees the left hand for document handling.

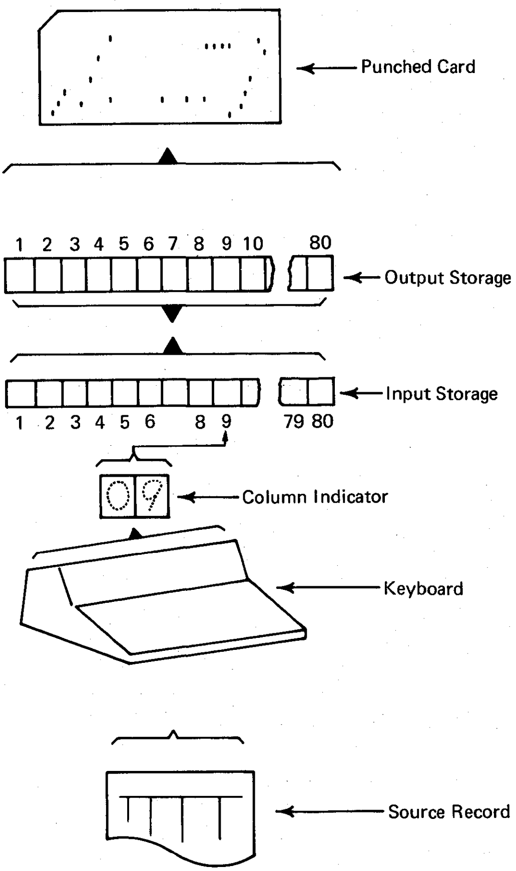


Figure 5.4 Data flow, buffered key punch.

The 129 Card Data Recorder contains input and output sections of data storage as shown schematically in Figure 5.4. Each storage section can hold electronically as many as 80 columns or characters of information from the source record. All data keyed for punching is first placed in input storage. A column indicator shows the operator the storage position where the next character is to be placed. After the source record is completely keyed into input storage, the information is transferred to output storage and then punched in a card. While one card is being punched from output storage, the next record can be keyed into input storage.

If a character is keyed incorrectly, the machine can be backspaced and the correct character keyed. Since a high percentage of such errors are recognized as they occur, the operator can correct most errors without spoiling a card. Rekeying erases the incorrect character in input storage and stores the correct character.

Program Control

Control features of the card punch can be made almost entirely automatic with the program feature. With this feature an operator can preset such functions as field tabulation or skipping, duplication of one field throughout the file, numerical or alphabetic keyboard shift, etc., before beginning each job.

The 129 is equipped with the capacity to store up to six different programs. Each program is set up by punching a specially coded card which exactly defines the fields and the operations to be performed. Using a special procedure, the operator then can read the program card into one of the six levels of program storage. The program mode switch is set to the desired level from 1 to 6. At any time while punching, the operator can select the appropriate program of control for a specific job or card format. This arrangement makes it possible to conveniently handle different record formats within the same file.

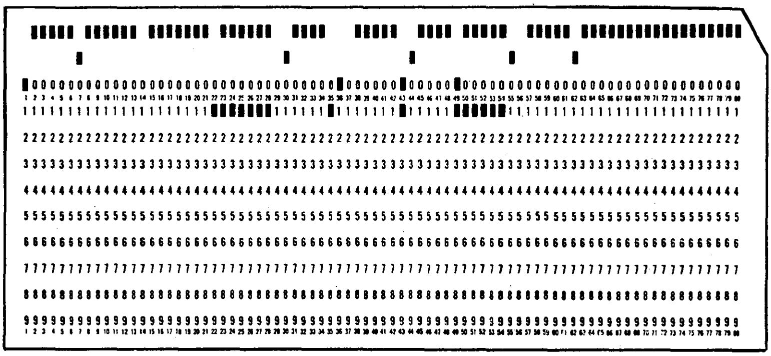


Figure 5.5 Punched program card.

Figure 5.5 is a punched program card. Codes for four basic functions are shown in the example, as follows:

Blank	Indicates the first column of a field to be manually punched from the keyboard into each card
1	Shifts keyboard to alphabetic mode for the column where it is punched
0	Starts automatic skipping
12	Defines the length of the field

Columns 14 through 21, 29, 37 through 42, and 56 through 61 are programmed for numerical punching. That is, if a combination key is depressed, a digit will be stored in input storage. Columns 22 through 28 and column 35 are programmed for alphabetic punching.

Columns 1 through 6 and column 36 of the cards being punched will be duplicated from the first card of the job. Duplicating and skipping are under manual control for the first card when the information to be duplicated is manually punched. Column 43 is also programmed to be duplicated, but only with alphabetic information or a blank.

Columns 7 through 13, 30 through 34, 44 through 48, and column 55 are skipped.

The preestablished machine control of record format greatly increases the efficiency of keypunching. When freed of the repetitively keyed functions of control, the operator can concentrate entirely upon the operation of actual data transcription. Figure 5.6 is a tabulation of all program codes for the IBM 129 Card Data Recorder.

Function	Punch Code	Field Location
Field Definition (FD)	12	All columns, except first column
Auto Skip Field	11	First column
Auto Dup/Auto Ver Field	0	First column
Alpha Shift (Programmed)	1	Each column for alpha shift
Self-check Field	2	First and last columns
Left-zero Field (Verify Only)*	3	First and last columns
Add Accumulator A	5	First column
Add Accumulator B	6	First column
Add Accumulator C	7	First column
Punch Accumulator A	4 + 5	First column
Punch Accumulator B	4 + 6	First column
Punch Accumulator C	4 + 7	First column
Punch and Reset Accumulator A	4 + 5 + 8	First column
Punch and Reset Accumulator B	4 + 6 + 8	First column
Punch and Reset Accumulator C	4 + 7 + 8	First column

Figure 5.6 Program codes, IBM 129 Card Data Recorder.

Other Features

Two models of the IBM 129 can print the character punched in a column along the top 12-edge of the card. The printed character corresponds to the punching. The printing feature can be operated under either manual or program control. Thus, in one operation cards can be prepared for both manual and machine processing.

A *self-checking number* feature provides for verifying precoded numerical information at the same time it is punched. This type of verification eliminates any need for later verification by other methods. This feature is described in more detail later in this chapter. The feature also operates under program control. An 11 punch is punched in column 81 of the data card to signify a valid self-checking number.

An *accumulate* feature provides the ability to balance to a predetermined total, to create a total for a group of cards or a batch of work, or to crossfoot and punch totals in the same card or in a following card, under program control.

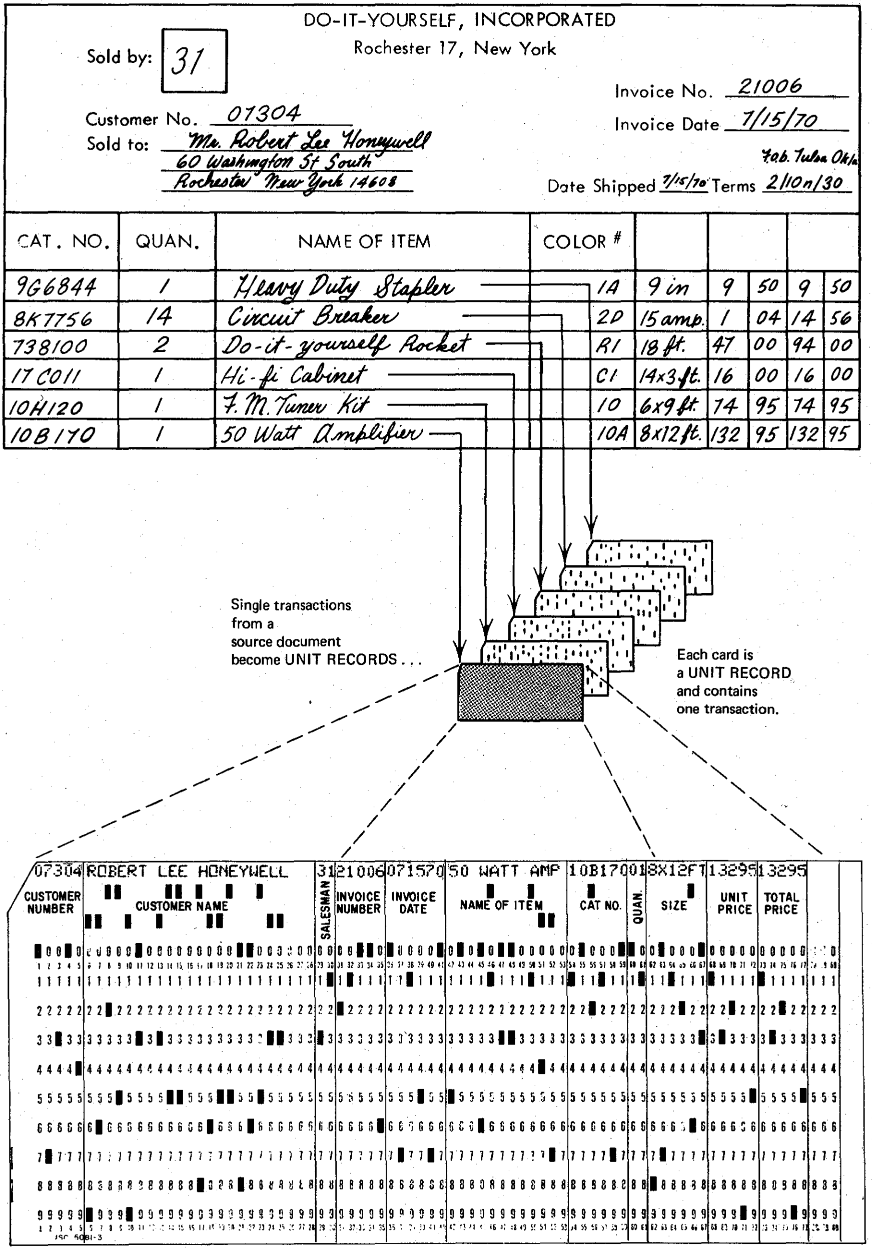
The machine is equipped with three 14-position, individual accumulators that add any numerical manually keyed or duplicated fields, in punch or verify mode, into any accumulator. The maximum size of the input field is 14 positions, thus any accumulation beyond the fourteenth position is lost.

Punchout, or punchout and reset, of individual accumulators are under program control, and may be done selectively. Group totals are normally punched out by way of a different program level than the accumulating program level. Amounts to be accumulated can be entered only by keying or duplicating.

A field will be subtracted in an accumulator if an 11 punch is multipunched in the units position of the field. Minus totals are punched out with an 11 punch over the units position of the field.

The *production statistics feature* provides statistics on machine production for use in measurement of work load, analysis of errors, and job accounting. A *keystroke counter* of six positions (000,000 to 999,999) counts every data keystroke in either punch or verify mode and functional keystrokes with some exceptions. A four-position *card counter* counts every output record in punch mode and every verify correct or correction punched card in verify mode. It does not count error cards in verify mode, program cards read in, and data cards read in. A four-position *verify correction keystroke counter* counts each verify correction data keystroke in verify mode.

All counters may be punched out, or punched out and reset by the operator. Totals may be obtained by batch, job, day, etc. All counters are reset to zero when the power is turned on to the machine.



DO-IT-YOURSELF, INCORPORATED
Rochester 17, New York

Sold by: 31

Invoice No. 21006
Invoice Date 7/15/70

Customer No. 07304
Sold to: Mr. Robert Lee Honeywell
60 Washington St South
Rochester New York 14608

f.o.b. Tulsa Okla.
Date Shipped 7/15/70 Terms 2/10n/30

CAT. NO.	QUAN.	NAME OF ITEM	COLOR #					
9G6844	1	Heavy Duty Stapler	1A	9 in	9	50	9	50
8K7756	14	Circuit Breaker	2D	15 amp.	1	04	14	56
738100	2	Do-it-yourself Rocket	R1	18 ft.	47	00	94	00
17C011	1	Hi-fi Cabinet	C1	14x3 ft.	16	00	16	00
10H120	1	F.M. Tuner Kit	10	6x9 ft.	74	95	74	95
10B170	1	50 Watt Amplifier	10A	8x12 ft.	132	95	132	95

Single transactions from a source document become UNIT RECORDS . . .
Each card is a UNIT RECORD and contains one transaction.

CUSTOMER NUMBER	CUSTOMER NAME	SALESMAN	INVOICE NUMBER	INVOICE DATE	NAME OF ITEM	CAT NO.	QUAN.	SIZE	UNIT PRICE	TOTAL PRICE
07304	ROBERT LEE HONEYWELL	31	21006	071570	50 WATT AMP	10B170	01	8X12FT	13295	13295

Figure 5.7 Transcription process from original record to 80-column cards.

The Transcription Process

Figure 5.7 is a schematic of the transcription process from an original record to punched cards. In this example, the record is an invoice that lists items sold to a customer. The example illustrates the following basic principles that should be considered in the design of any key punching transcription procedure:

1. The original record always contains *historical* data. For example, it is assumed here that the quantity of goods itemized are being delivered or have already been delivered to the indicated customer at specified prices.

2. Historical records almost invariably originate in a location at some distance from the keypunching operation. It is unusual for the keypunch operator to take any part in the preparation of a record. In fact, the operator most likely has little or no knowledge of the conduct of the business involved or of the characteristics of the goods or services to be punched as data in cards.

3. The operation of keypunching must always be reduced to one of *simple transcription.* An operator cannot be expected to decode, rearrange, or translate data from original record to punched card.

4. Conversion of identification or other data to numerical codes before the document arrives in the keypunching department greatly reduces the number of holes to be punched and more efficiently uses the 80-column capacity of the card.

The invoice in Figure 5.7 has been coded in advance for keypunching. Numeric codes have been assigned to the items of sales office and salesman, city, state, and customer number. Note that this coded data is common to all items listed on one invoice. As other unit record processing operations are reviewed, it will be seen that the handling of numeric data is simpler and more efficient than the handling of alphabetic data. This is a further advantage of reducing information to numerical data whenever practical.

5. The format of the card is arranged to accommodate data in the same sequence as it is read from the invoice by the operator. Customer name, invoice number, date, and sales identification are in the left end of the card (the first columns under the punching station). This information is punched only once into the first invoice card and is then duplicated into each succeeding item card.

6. The card record is logically divided into its component fields, each a separate item of data within the record. The amount of information to be transcribed should be held to the minimum required for final reporting as required by the procedure. Field size is designed with the proper number of columns to contain the maximum number of characters in the field.

7. The accuracy of the transcription process must be controlled. The first concern is to assure the accuracy of key depressions and punched holes. Aside from the normal handling and accounting controls, the primary method is one of machine verification. This process is described in the next section.

The Machine Verification Process

The machine verification process is intended to be an exact repetition of the original transcription operation from written documents to punched cards. The purpose of verification is to assure the accuracy of transcription. The following considerations should be noted:

1. The verifying operation, like that of keypunching, is strictly mechanical. The operator of a card verifier duplicates exactly the reading, keying, and handling of both documents and cards. Whenever possible, work should not be punched and verified by the same operator.

2. Normally, all transcribed data need not be verified. In many applications, descriptive information such as name, size, weight, color, etc., is not essential to the accounting accuracy of the procedure. For example, even if the name of an item is misspelled, chances are the item can still be identified properly on printed reports. As in all procedures, the value of assuring proper spelling of names must be measured against the cost of machine verification.

On the other hand, if customer names and addresses are involved, it may be well worth the extra cost of verification to avoid the embarrassment and possible loss of business that could occur if shipment is made to a wrong address.

3. Mechanical verification does not assure *absolute* accuracy. A verifier operator *can* misread the original document in the same way the keypunch operator did. Or, the original document may be illegible or confusing, making transcription accuracy unreliable. Other methods of assuring accuracy are discussed in a later section.

4. It must be stressed again that proper design of documents and card forms is essential for accurate, efficient card punching and verifying. When any good operator is observed in action, it is immediately apparent that rhythm plays an important part in achieving efficiency. Just as a good typist falls into an easy, fast rhythm of word keying, carriage return, and paper handling, the skilled card machine operator works best with a good rhythm of card feeding, keying, skipping, duplicating, and document handling. Even the sound of the machine in operation indicates to the experienced operator that the work is being done accurately; particularly that data fields are in the proper columns.

96-Column Card Punching and Verifying

The 96-column card is the principal data entry medium for the IBM System/3. As a completely independent unit in this small computer system, the IBM 5496 Data Recorder does the jobs of

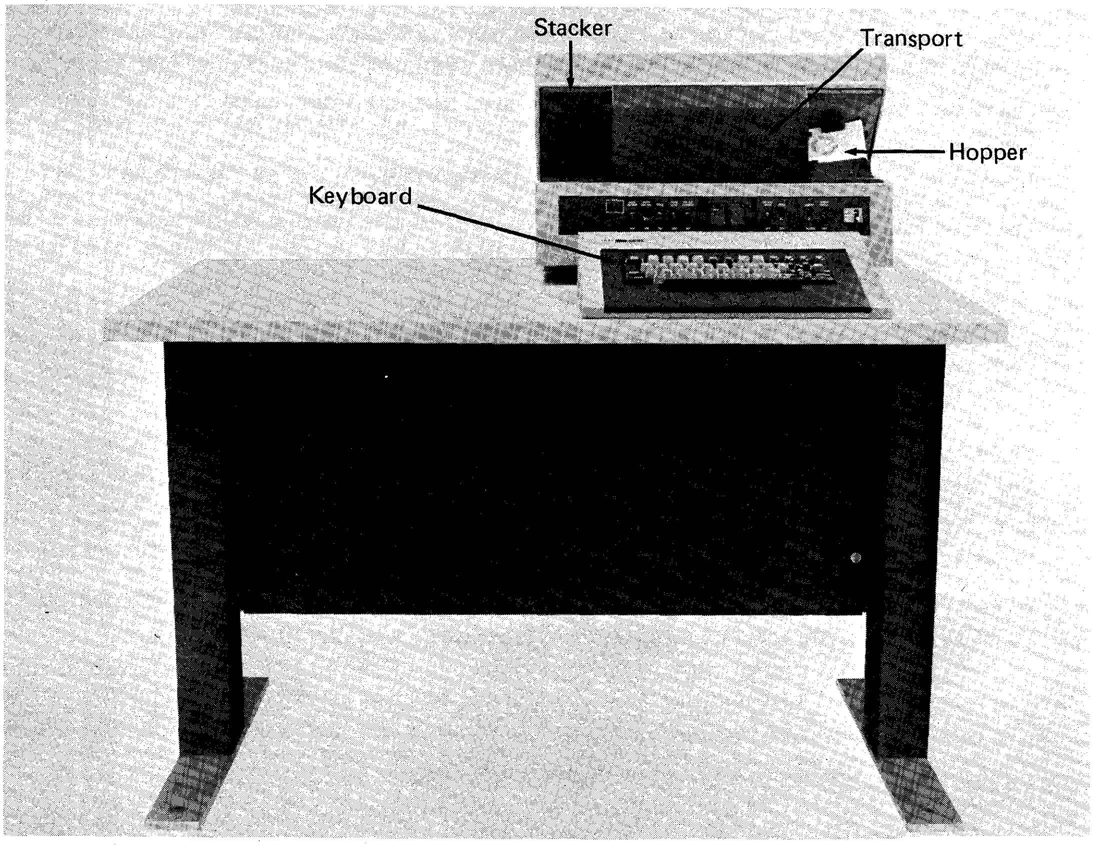


Figure 5.8 IBM 5496 Data Recorder.

both creating this latest form of unit record and verifying data which has been previously recorded. In addition to combining the two functions in one machine, the 5496 Data Recorder has been designed with a number of improvements over the older 80-column equipment.

Compactness is made possible by the use of smaller cards, as shown in Figure 5.8. Also, the unit has the capability to *store* as many as four levels of control programs at the same time. This complements the greater flexibility of the stored program System/3 to process multiple record formats.

Punching

All keyed characters are placed first in *entry storage.* This operation is very similar to that of the 129 80-column punch. Keying is carried on without punching, until all data to be placed in a single card is stored. Then, by pressing a release key, the stored data is transferred to *output storage,* from where the entire card is punched automatically. Punching is done at a speed of 20 columns per tier per second. Three columns are punched at a time, one in each of the three tiers; for example, columns 1, 33, and 65 or 17, 49, and 81. The result is a punching (and printing) speed equivalent to 60 characters per second. Data for the next card can be keyed into entry storage while the preceding card is being punched. When data enters storage, it replaces any information previously stored there.

With the Data Recorder, provision is made also to *erase* a keying error from storage before the release key is pressed. Because no holes have been punched at this time, erasure can be made for a single column, a field, or for an entire card record. Release of data in output storage is entirely under the operator's control.

Operating Features

Some of the more important operating features are reviewed here.

The *card hopper*, with a capacity of 350 cards, is located to the operator's right. Cards are placed in the hopper printed side forward, punching positions toward the bottom. Cards are fed one at a time into a *transport* mechanism (Figure 5.9) where they are carried to punch, read, and print stations. Movement of the cards is from the operator's right to left.

The *stacker* at the left holds the completely processed cards in the same sequence in which they are punched or verified.

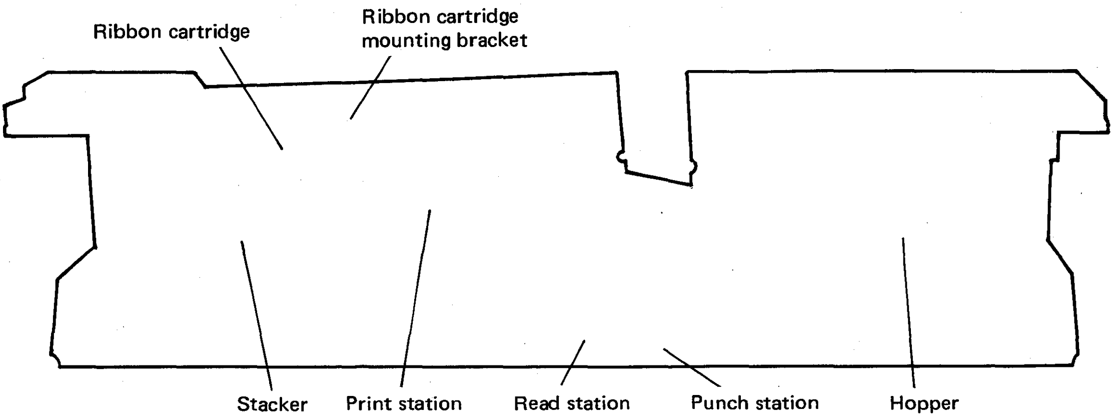


Figure 5.9 IBM 5496 Transport.

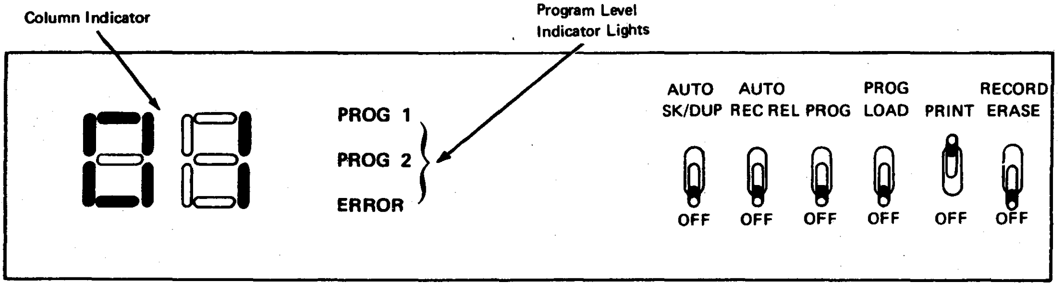


Figure 5.10 5496 operator's panel.

An *operator's panel* is located just below the transport mechanism. Figure 5.10 is a schematic of available control features on the panel. Note the *column indicator,* a lighted digital display that indicates to the operator the position of the *next* keyed character entry into storage. The position indicated corresponds to the card column in which that character will be punched. While operating the machine, the operator is not concerned with the actual punching or reading position of the card. The cards are not visible to the operator as they move through the transport mechanism.

Punching of a record begins when the column indicator advances from 00 to 01. At this point, information for the next card record can be keyed. The digits 01 through 96 are displayed in the indicator.

The combination *keyboard* rests on a flat desk at the front of the machine. Data keys are arranged in the same positions as the keys of 80-column machines with the facility for recording either alphabetic or numeric information. Control keys are positioned in each of the four rows. Figure 5.11 is a schematic of the Data Recorder keyboard.

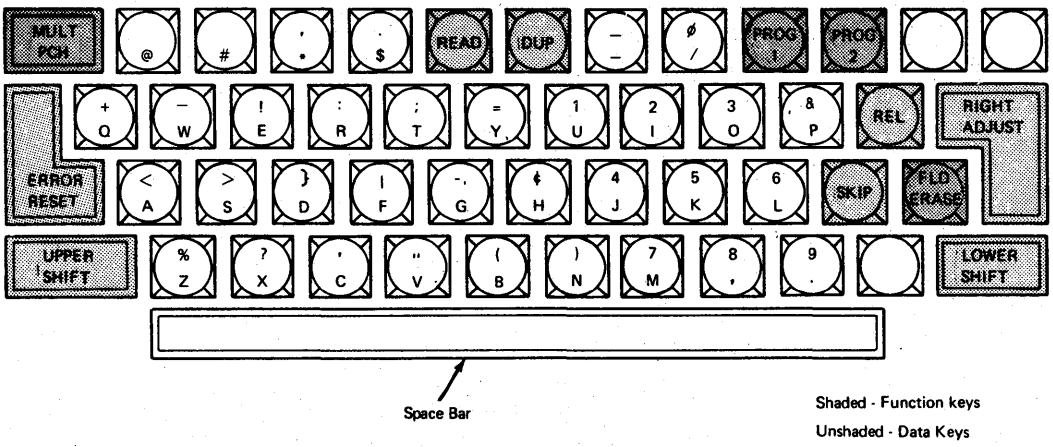


Figure 5.11 5496 keyboard.

Program Control Switch

When the PROG switch is on, the Data Recorder operates under program control. Information can be stored in four program levels that can be used to control automatic and manual punching functions.

When the program control switch is turned from off to on, the information in program level 1 is automatically in control. A different program level may be put in control by pressing another program level key on the keyboard: PROG 2, PROG 3, or PROG 4.

While keying data, it is possible to shift between program levels any number of times if the Data Recorder remains under program control. If a different program level key is pressed after the first column of a field has been passed, the program level in control automatically changes when the column indicator advances to the first column of the next field. This protection feature prevents changing inadvertently from one program level to another in the middle of a card field.

When the program control switch is off, the Data Recorder operates under manual control. Information is keyed on a column-by-column basis, disregarding any automatically programmed functions.

Program Card

A program card is prepared first by the operator with the Data Recorder under manual control. Punched hole codes are assigned to operate the various control functions of the machine. These codes are shown in Figure 5.12.

Description	Code	Keyboard Character to Press	Card Column Where Punched
End of Field	B	- (minus)*	Last column of every field
Automatic Skipping	A	Ø (zero)* *	Each column to be skipped
Automatic Duplicating	8	8	Each column to be duplicated
Lower Shift	4	4	Each column necessary
Numeric Shift	2	2	Each column necessary
Self-check Number Verification	1	1	First column of field only

When operating under program control, if a column is not coded for either lower shift or numeric shift, it is treated as upper shift.

If a column is not coded for either automatic skipping or automatic duplicating, it must be manually keyed.

* The minus (-) will produce the "B" punch position code.

** The zero (Ø) will produce the "A" punch position code.

Figure 5.12 5496 control codes.

For a control card to function, it must be read and stored in a program level storage area. The *program load switch* is used for this purpose. Four different storage areas are provided and each level can store information from one program card at any given time. As part of the loading procedure, the operator selects which storage area is to receive the program. The program card is placed in the hopper, fed through the machine, and read into storage at the read station. Once placed in storage, the program remains there until replaced by another program.

When a keypunching job is in process, the operator can select any one of the four stored programs to control operation.

Automatic Duplication

Information common to a given number of cards can be duplicated automatically by the Data Recorder. The data is read from the output storage area under either manual or program control. This is equivalent to reading data from the preceding card into the card being punched.

An auxiliary duplicating feature is also provided. Information is read first into the fourth program level storage from a punched card in the normal manner. Whenever this information is to be duplicated, it can be placed in key entry storage by the operator and from there into the output card.

Right Adjust

This function is supplied by the *left-zero feature* of an 80-column card machine. It operates only under program control when the end of the field has been defined by an end-of-field code (hyphen) in the program.

When the right adjust key is pressed, keyed characters in the field are shifted to rightmost columns. The insignificant positions to the left of the field are left blank for alphabetic fields and are punched zero, without keying, for numerical fields. Right adjusting can thus be done with either alphabetic or numerical fields.

Figure 5.13 shows the source document to card relationship of the 96-column card. Figure 5.14 is a program card prepared for this particular punching procedure.

96-Column Card Verification

The Data Recorder can be used as a verifier by setting the mode switch on the operator's panel to *verify*. Cards to be verified are placed in the hopper in the same sequence as punched. Cards are read one at a time into the output storage area. This is done by pressing the read key.

Each card is stopped at a *verify wait station* after reading. The operator then reads the same data from the source document and keys the data a second time. As each character is rekeyed it is automatically compared with the character in the same column in the output storage area. When the comparison agrees, keying can continue until the entire card is verified. If the verification is completed without error, a notch is punched in the trailing edge to mark the card as correct. The verification notch is shown on a card in Figure 5.15.

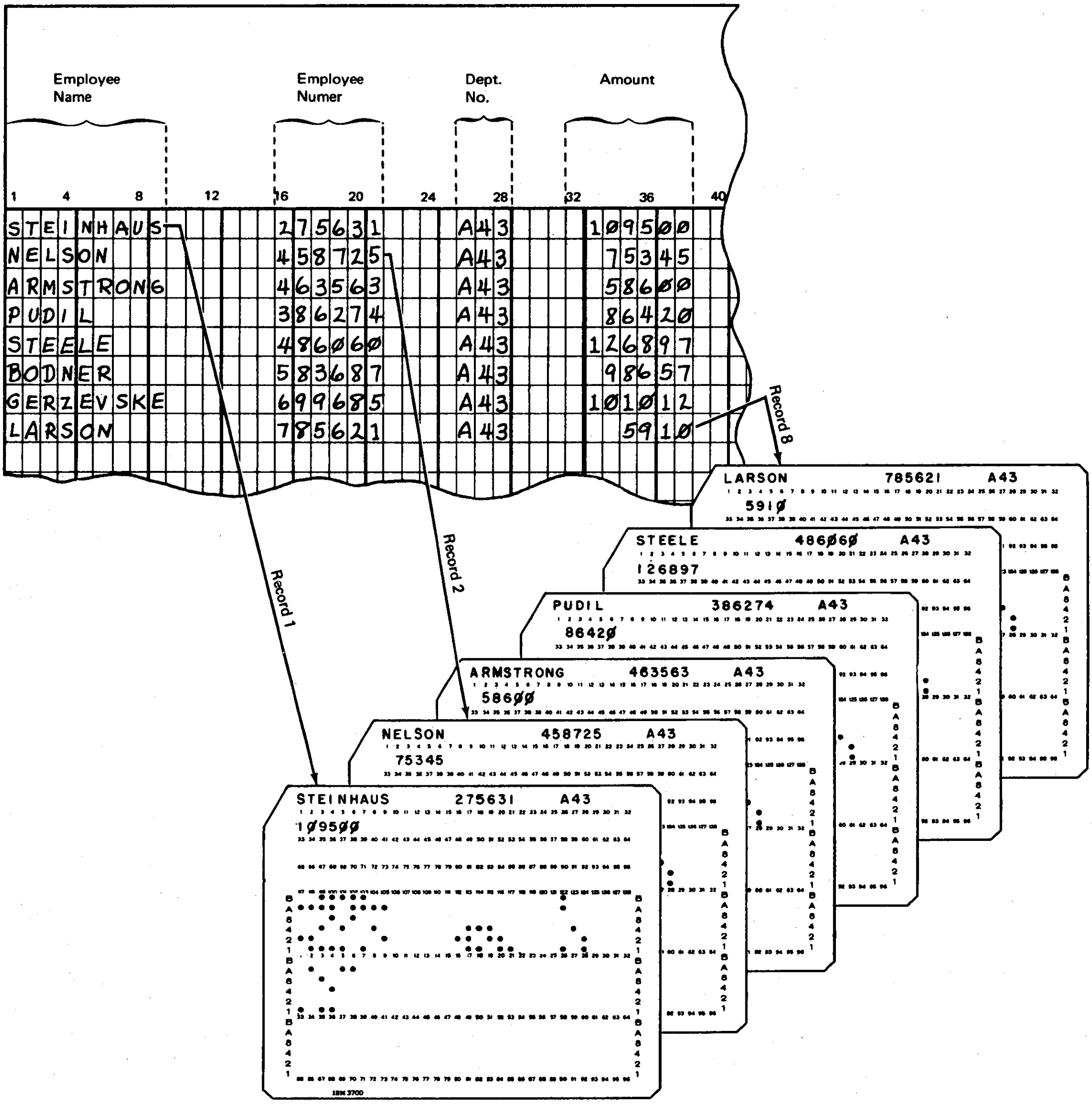


Figure 5.13 Source document to 96-column card.

If the character keyed does *not* agree with the character for that column in storage, an error light turns on, and the keyboard locks. The operator must find the error before keying the next character. At this point it is known only that the comparison does not agree. The operator can turn off the error light, free the keyboard, and try again. If the verifier made the error, it is presumed the correct key would be pressed on the second try. If this is done, verification can continue.

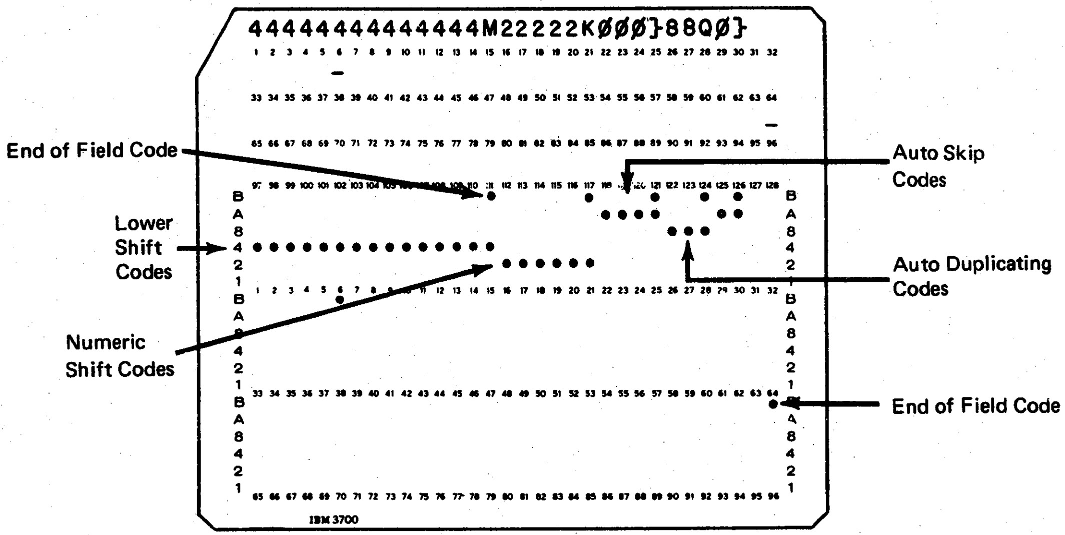


Figure 5.14 Program control card for sales report.

A total of three tries is permitted. If the character comparison between output storage and the keyed character *still* do not agree, it is presumed the error was made by the keypunch operator and that the wrong character is punched in the corresponding card column. Therefore, a character keyed for the third time is automatically stored in the output storage area, replacing the incorrect character.

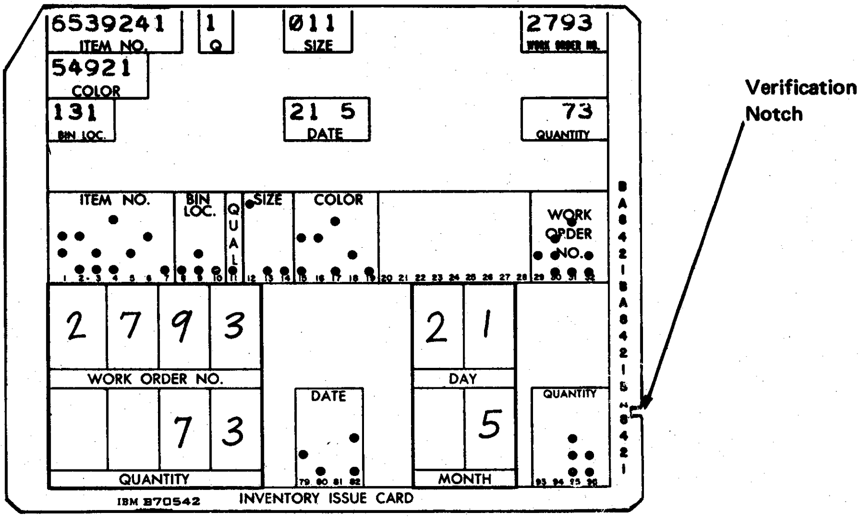


Figure 5.15 96-column card verification notch.

A correct card can be repunched at this time from the data in the output storage area. The operator inserts a blank card in the hopper and turns on the *verify repunch* switch on. This causes the card containing errors to be stacked, the corrected record in the output storage area to be punched (and that card notched) and the next card to be read for verification. With this procedure, error cards can be immediately withdrawn from the deck at the time of verification. Correct cards can be inserted by the operator. At the completion of the verification procedure, the deck of cards from the Data Recorder should be complete and ready for processing.

Questions and Exercises

1. Name the three essential elements of the data processing system.
2. What is the difference between real time and historical data? List ways in which each type can be represented.
3. Explain the advantages of the "buffered" key punch.
4. What is the function of the card punch program?
5. List some additional useful features of a card punch other than the function of coded hole punching.
6. Summarize the main points of the transcription process.
7. Discuss the advantages versus the disadvantages of card verifying.
8. Lay out the essential elements of an itemized invoice for general merchandise.
9. Design an 80-column card for transcription to machine records from the invoice.
10. Design accounting controls for the transcription operation.
11. Sketch out a procedure for the flow of work.

6 PUNCHED CARD DESIGN

To quite a large extent, the successful installation of a data processing procedure depends upon proper forms design. This is true regardless of whether the system is to be carried out by automated or manual methods. Forms must be carefully developed to suit each particular need in order to make work flow as conveniently and efficiently as possible. This chapter discusses punched card design features of both 80- and 96-column records.

Types of Punched Cards

Consideration of card form design usually begins by determining how the document is to be originated. One or more of the following types of cards can be adapted to fit almost any kind of machine data processing procedure.

Transcript cards are punched directly from information recorded on other documents. The information is always historical, that is, it represents transactions that have occurred at some time before transcription takes place. In earlier sections, the card forms shown to illustrate the operations of data transcription are examples of this type of card design. Other common examples include cards punched from invoices, vouchers, receipts, orders, inventory listings, and detail item cards of all sorts.

Many master cards fall in this category: cards punched from credit card applications, personnel records, customer name and address files, item description and price cards, and so on. Master cards are normally transcribed once to set up the file and are then used repeatedly with batches of detail transactions that originate from other source documents.

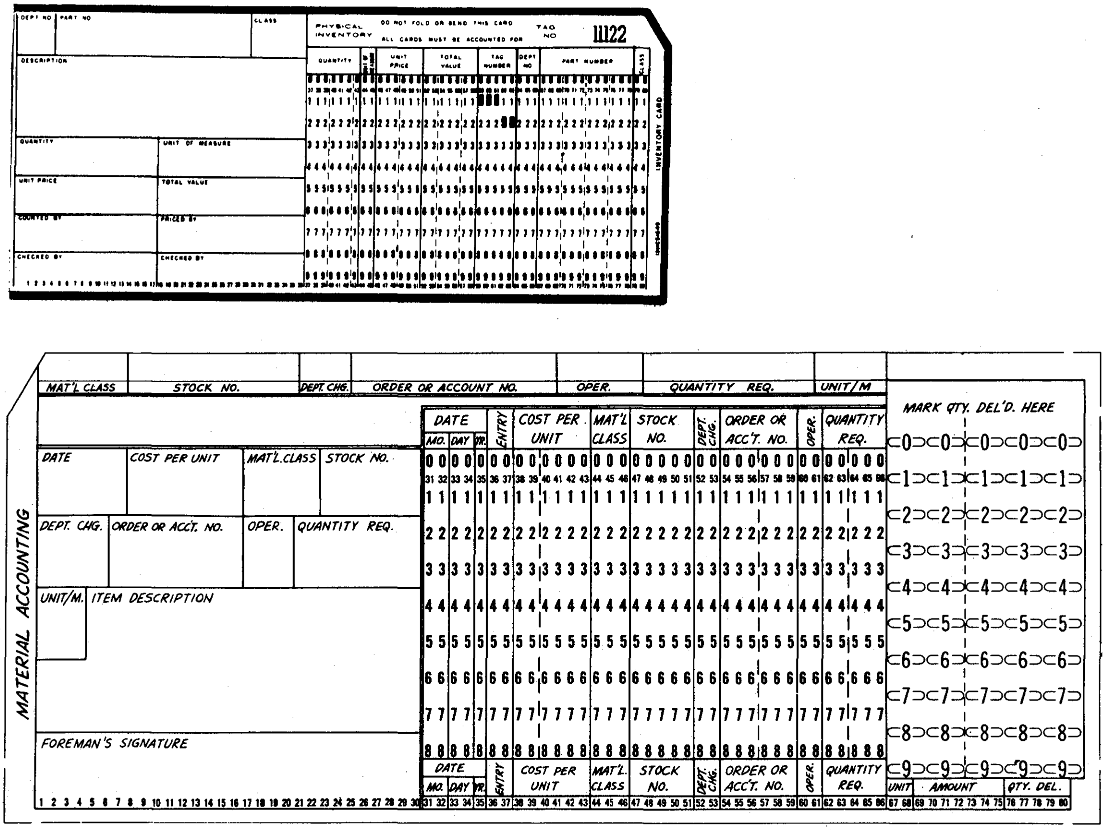


Figure 6.1 Dual card designs.

Dual cards are punched from information previously written on the cards themselves. Figure 6.1 is an example of this type of card. Such cards serve as both source documents and processing media. Examples include:

1. Inventory accounting and recording cards. The quantity of an item in stock can be handwritten on the card when physical inventory count is taken in the warehouse. Identification or part number may also be handwritten with some description of the item. Cards are sent to the keypunch department for

86417

punching of the written data; they are later used in processing an inventory report.

2. Payroll change cards may have the change written directly on the card, for example, a deduction authorization to meet some obligation which is "payroll deductible." The written information identifies the type of deduction, the amount, when it is to be paid, employee identification, and also usually includes the employee's signature of authorization. After processing, the card becomes a permanent record of authorization to pay the obligation. Payroll change cards may also be authorized by an employer for hourly pay rate or salary changes, work shift changes, or job assignment changes.

3. In production and inventory control procedures, many dual cards are originated as requisitions; that is, requests to withdraw an item from stock for production purposes in a factory.

The dual card is the most versatile of all the punched card record forms. It has the advantage of enabling handwritten information to be processed by machine methods. Both original and punched hole information are permanently recorded in one medium.

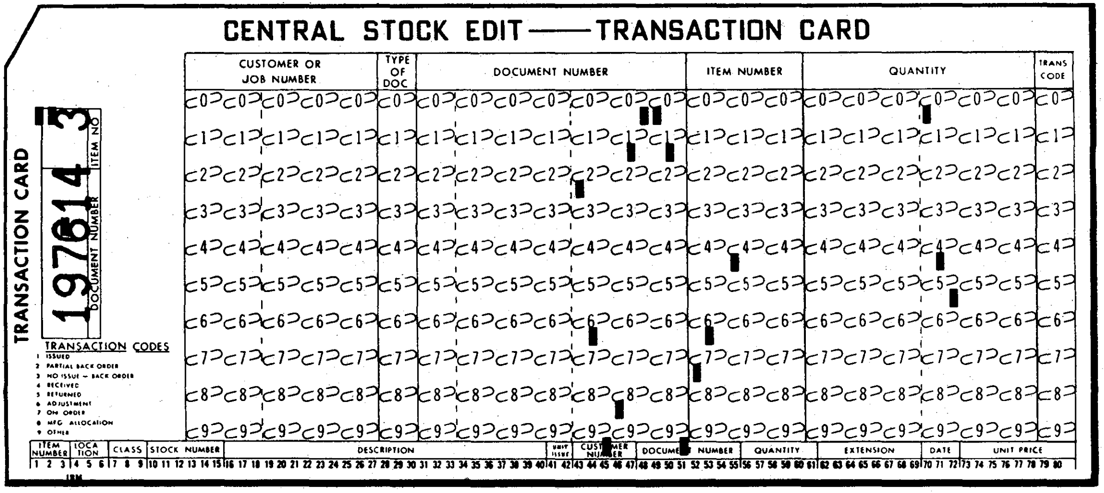


Figure 6.2 Mark sensed card design.

Mark-sensed cards are automatically punched from pencil marks that are made in predetermined positions on the face of the cards. Figure 6.2 is an example of this type of unit record. The pencil marks record data at the source of the transaction or event.

The marks can be machine read and punched into the same or a different card for processing.

Examples include weekly attendance records marked by a timekeeper and meter readings made by employees of a utility company. The most practical use of the mark-sensed card is to record quantities of items on hand, in stock, sold, moved, received, and the like. The card should be marked "on location" at the time of the transaction. The size of the mark-sensed field is usually kept to 10 positions or less.

Output cards are automatically created as a result of some machine processing. This type of card is punched as a summary, balance forward, file update, or exception card by other unit-record equipment. These operations will be discussed in more detail in later sections.

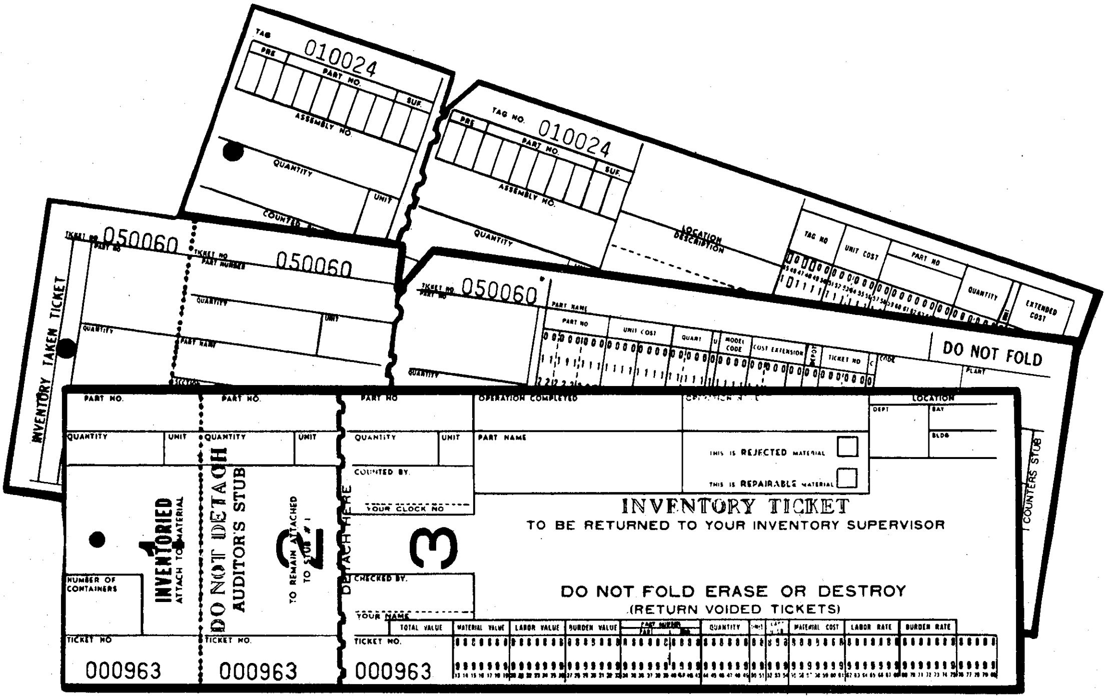


Figure 6.3 Stub card design.

Output cards can also be punched by computers and computer peripheral equipment. The cards may be created as intermediate records in a procedure, serving to hold intermediate results of calculations which will be discarded later. On occasion, source record dual cards are copied or *reproduced* by machine in order to preserve the original file intact while processing is done on information in the duplicate file.

Special purpose cards take many forms, some of which are shown in Figure 6.3. Manufacturers can provide *scored* cards that can be pulled apart, leaving a shorter card record of 51, 60, or 66 columns. Some 80-column card machines can be especially equipped to handle these shorter card lengths.

One of the most commonly used forms of special 80-column cards is the punched card check. Checks usually originate as output cards from a payroll procedure. That is, the check may be a summary card of the period earnings after all calculations of pay and deductions have been completed for each employee.

Next, the check is *interpreted*; that is, the information punched is printed on the card to show payee and amount. The check is then made valid by affixing the authorized signature of the payer. When the check is cashed, the payee endorses the card on the back as would be done with any other type of check. The card can then be machine processed by the bank and eventually by the payer to reconcile his bank account.

Determining Card Data

The type of card record to be used will be determined by the requirements of the procedure. Sufficient data must be obtained to produce the desired results. The following should be considered:

1. *Source information,* to be transcribed from original records or punched in dual cards.

2. *Constant data,* or such information as price lists, job codes and wage rates, descriptions, names, sizes, customer names and addresses, and other coded data which must be inserted to complete the card records. In most procedures the constant data is gang-punched or reproduced from master card files.

3. *Calculation results.* For example, to work the formula *Quantity* × *unit cost* = *total cost* the *quantity* is obtained from the source document as a transaction, *unit cost* from the price list master file, and *total cost* as a product of a multiplication operation. Space must also be provided to punch any intermediate results that may be obtained in a calculation but which do not necessarily appear in the end report.

4. *Reference data* should be adequate to identify each transaction with the source document from which it was created and include the date on which it occurred. In some cases the data should designate a person, place, or item responsible for or involved with the transaction. Examples include date, invoice number, batch number, salesman number, etc.

It is nearly always a requirement of the procedure to identify the different types of cards by punched codes. This makes it possible, for example, for machines to distinguish between master and detail cards or between transcript and dual cards.

The amount of reference data required will be determined by the use to which the cards will be put and the reports to be written from them.

5. *Machine produced data.* Other than calculation, certain forms of data can be emitted by the equipment with the use of special devices: digit emitters, storage, and so on. Such data is sometimes required to obtain intermediate results.

Determining Field Size

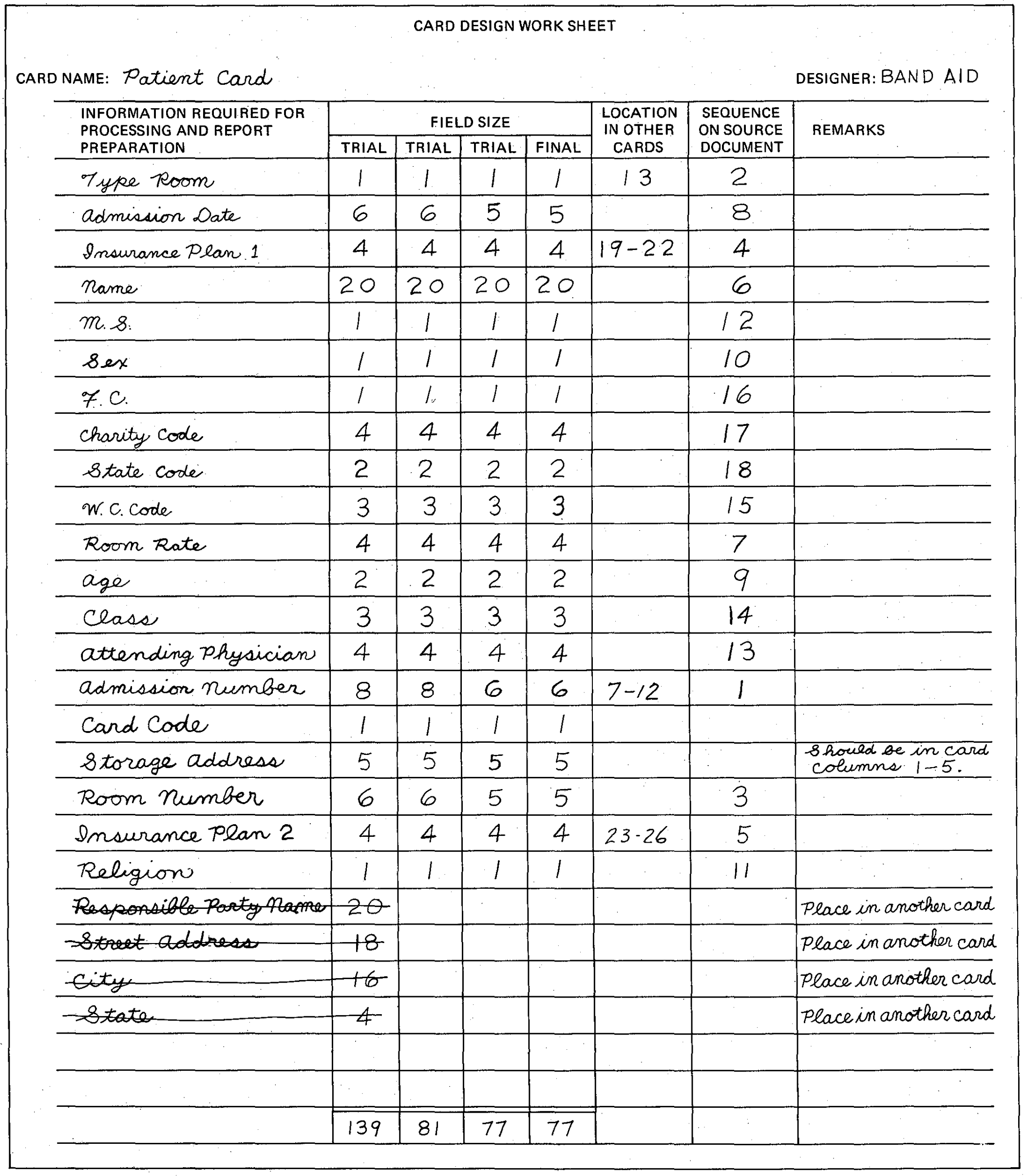
The card design work sheet shown in Figure 6.4 is one way to organize information to determine card field size and arrangement.

The number of columns required to record each type of information is listed. For codes, date, invoice number, account number, etc., this is determined by the largest single number to be recorded. Thus, four columns are needed for invoice number if the number series is to be repeated after 9,999; two columns are needed for office if there are no more than 99 branches.

For quantity and amount fields, the problem is slightly more difficult. In the first place, the space needed to record the largest amount may not be known. Second, if the largest amount is known, its chances of occurring may be rare.

It is therefore a good plan to provide for all except those rare cases, and handle them by punching extra cards. For example, if the seven-digit amount $67,265.80 is a rare case, it may be recorded in a six-column field by punching seven cards for $9,000.00 and one card for $4,265.80 or any convenient combination of cards totaling $67,265.80. In designing summary and balance forward fields, however, the amount fields must be made large enough to accommodate the summary totals.

For punching names, 20 columns are usually sufficient. The requirements must be checked on each individual job. As a general



CARD DESIGN WORK SHEET

CARD NAME: Patient Card

DESIGNER: BAND AID

INFORMATION REQUIRED FOR PROCESSING AND REPORT PREPARATION	FIELD SIZE				LOCATION IN OTHER CARDS	SEQUENCE ON SOURCE DOCUMENT	REMARKS
	TRIAL	TRIAL	TRIAL	FINAL			
Type Room	1	1	1	1	13	2	
Admission Date	6	6	5	5		8	
Insurance Plan 1	4	4	4	4	19-22	4	
Name	20	20	20	20		6	
M. S.	1	1	1	1		12	
Sex	1	1	1	1		10	
F. C.	1	1	1	1		16	
Charity Code	4	4	4	4		17	
State Code	2	2	2	2		18	
W. C. Code	3	3	3	3		15	
Room Rate	4	4	4	4		7	
Age	2	2	2	2		9	
Class	3	3	3	3		14	
Attending Physician	4	4	4	4		13	
Admission Number	8	8	6	6	7-12	1	
Card Code	1	1	1	1			
Storage Address	5	5	5	5			Should be in card columns 1-5.
Room Number	6	6	5	5		3	
Insurance Plan 2	4	4	4	4	23-26	5	
Religion	1	1	1	1		11	
~~Responsible Party Name~~	~~20~~						Place in another card
~~Street Address~~	~~18~~						Place in another card
~~City~~	~~16~~						Place in another card
~~State~~	~~4~~						Place in another card
	139	81	77	77			

Figure 6.4 Card design work sheet.

rule, 95% of the names of individuals can be recorded in 18 columns or less; 90% of all street addresses require 18 columns or less; 95% of the names of companies require 20 columns or less; and 99% of all cities and states (abbreviated) require 20 columns or less.

The total number of columns required for all fields will show whether the capacity of the card has been exceeded (80 or 96 columns). If the number of columns must be reduced, the following techniques may be used:

1. The size of reference fields may be reduced by repeating a numbering series more frequently. For example, invoice numbers may start with 1 each month instead of each quarter.

2. Reduce the size of control fields by making certain fields subclasses of others. For example, two digits may be assigned to branch office code and three to salesman code. Branch code may not be needed if salesman code is assigned sequentially by branch; that is, salesmen 001 through 005 are assigned to branch 01, salesmen 006 through 010 to branch 02, salesmen 011 through 015 to branch 03, and so on. In this series, the salesman number also designates branch, although the coding is slightly more difficult to handle in the processing.

3. Reduce the size of reference or control fields by recoding. It is often possible to eliminate several columns.

4. Reduce the number of columns required for recording reference or control fields by ignoring one or more digits which may not be essential. Thus, it may be possible to punch only four digits of a six-digit invoice number and retain positive identification.

5. Reduce the size of the amount fields in those cases where the number of digits in the amount seldom exceeds the capacity of the field.

6. Record in the 11th and 12th punching positions of the 80-column card information which is never used for printing. These positions can be easily used in columns that are set aside for multiple punching. However, they should not be used in control or alphabetic fields.

7. Use a single card column for recording several one-digit codes. For example, if an applicant is male, use an 11 punch; if female, use a 12 punch. Age groups can be designated by a 0 punch for ages 15–20, a 1 punch for ages 21–23, a 2 punch for ages 24–26 ...and a 9 punch for age 65 and above. In this way, two categories of information can be punched in the same column. Since this is multiple punching, caution should be exercised that the different combinations are machine readable on 80-column equipment. If a

96-column card is being designed, any one of 64 different characters can be punched in a single column.

8. Avoid unnecessary data. For example, the use of both an order number and an invoice number is redundant if one field will provide adequate reference.

9. Use one field instead of two when alternative information is to be recorded. An example is an accounts receivable transaction card that is to be used to record data from either an invoice or a credit memorandum. The use of two fields, Invoice Number and Credit Memo Number, is unnecessary if an 11 or digit punch in another column of the card identifies the Credit Memo. The 11 punch is usually placed over the amount field to indicate a subtraction from the customer's account. The field can now be labeled Invoice or Credit Memo Number.

10. For cards that are to be used for statistical analysis, combine several classes of data into a single column. This technique is especially applicable to yes/no responses.

If it is necessary to use more than one card, the data are separated or classified to determine which should be placed in each card. Such a division may be based upon any one of several schemes:

a. Place the repetitive or recurring information in one card and temporary or nonrepeating information in a second card, as in the case of master cards and detail cards.

b. If more than one source document is involved, design a different card for each and use card codes.

c. In cases where one transaction affects two different accounts, design two cards, each with a different degree of detail. Examples are accounts payable and payables distribution cards, accounts receivable and sales cards, payroll and labor distribution cards.

d. For printing invoices, orders, etc., design a card for each section of the form. For example, a job may include for each customer (1) header cards, (2) a miscellaneous data card, and (3) detail commodity cards. With such an arrangement, the heading and miscellaneous data cards may also be used for other jobs such as printing credit memorandums and packing slips.

Determining Data Sequence

There are four major considerations in determining a card's data sequence.

1. The location of identical data in other cards with which the new one is to be processed

2. The sequence of data on the source document from which a transcript card will be punched

3. The machines and programs to be used in the processing

4. The manual operations in which the card will be used

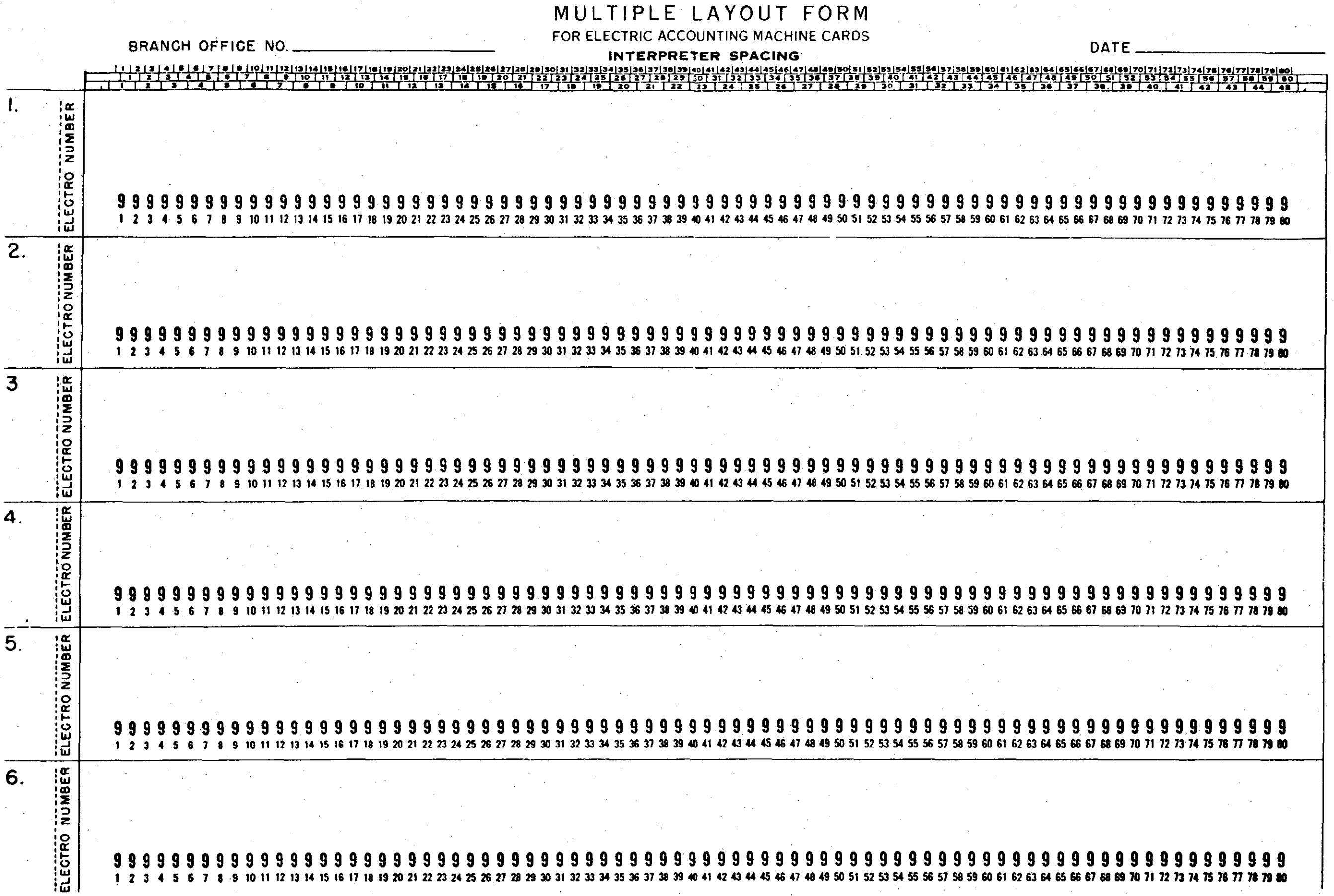
Specific areas have been made available on the card design work sheet (Figure 6.4) for recording the location of data in other cards as well as the sequence of data on the source document. The effect upon sequence of machines and programs as well as manual operations and the location of printed information can be indicated in the remarks column. The completed card design work sheet should be quite useful in later design operations of machine processing procedures.

Location of Data in Other Cards

A given field of information in a new card should be placed in the same columns previously assigned to it in other cards. This assures that sorting and controlling can be accomplished when the cards are processed together. It also makes control panel wiring and programming easier. As will be shown later, these are means of setting up and controlling equipment to do a specific job. In cases where summary cards are designed to accommodate year-to-date figures or balance forward amounts which by their nature must be larger than corresponding fields in the detail transaction cards, the amount fields in both should be aligned as nearly as possible.

Examples of different cards which are processed together include (1) a customer name card used with accounts receivable cards to write a statement, (2) a daily time ticket used with labor distribution cards to obtain a zero balance, and (3) a labor distribution card with material distribution cards for cost analysis.

Figure 6.5 is a multiple card layout form that can be used when planning the design of several cards at one time or when planning a new card to be used with existing cards. The form helps align those fields that are common to more than one card.



MULTIPLE LAYOUT FORM

FOR ELECTRIC ACCOUNTING MACHINE CARDS

BRANCH OFFICE NO. ____________ DATE ____________

INTERPRETER SPACING

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45

1. ELECTRO NUMBER

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

2. ELECTRO NUMBER

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

3 ELECTRO NUMBER

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

4. ELECTRO NUMBER

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

5. ELECTRO NUMBER

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

6. ELECTRO NUMBER

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Figure 6.5 Multiple-card layout form.

After all fields have been laid out and sequenced, a separate form is available for designing each card in detail. This form will be discussed later.

Sequence of Data on the Source Document

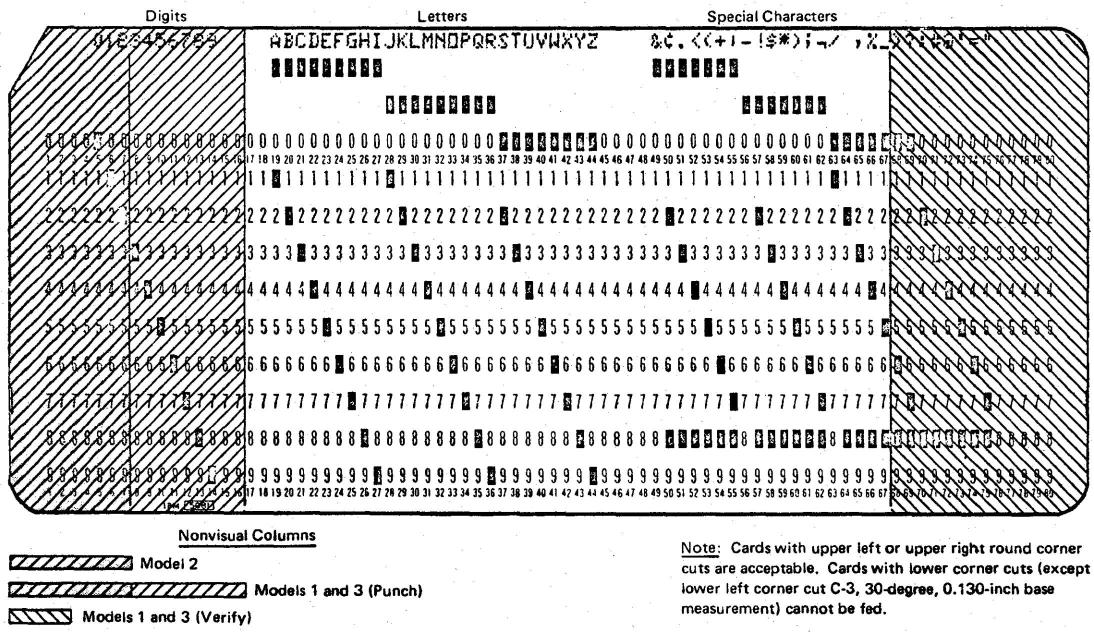
Keypunching, as a manual operation, is subject to a considerable variation rate. Anything that simplifies it will tend to produce a faster and more accurate operation. Data should be punched into a card in the same order in which it is read. If the data sequence on the card is considerably different from that on the source document, it may be necessary to redesign the source document. There is sometimes greater flexibility in the arrangement of the data on an original document than on the card record which is to be machine processed.

Machine Characteristics

Cards can be designed to take full advantage of both machine characteristics and processing techniques. Although equipment operation has not been discussed yet, the logic of some of the following considerations should be apparent as good practice in all accounting procedures.

Card Punches

In designing a dual card, it is better to locate the handwritten data towards the left end and the punched data on the right. This keeps the handwritten information from being hidden under the card machine pressure lever while the card is read for punching. Figure 6.6 shows what portion of the 80-column card can be covered when the card is in the bed of the 129 Card Punch. When other equipment is to be used, the card design should be checked against the manufacturer's recommendations. Figure 6.7 shows standard and dual 96-column card visibility as these cards would appear in the hopper of the 5496 Data Recorder. The shaded areas represent portions of the card that are covered by machine parts. Figure 6.8 shows dual and standard card forms at the verify and wait station of the Data Recorder. As shown by the shading, a larger area of the card is covered here. If data keyed into a dual card is to be verified, the portion of the card with handwritten source data must be visible to the operator at this station.



Note: Cards with upper left or upper right round corner cuts are acceptable. Cards with lower corner cuts (except lower left corner cut C-3, 30-degree, 0.130-inch base measurement) cannot be fed.

Figure 6.6 Card visibility at punch station, IBM 129 Card Data Recorder.

Sorters

For sorting operations when cards are being sequenced by several control codes (e.g., branch, salesman, and product), operator control is simplified if the fields are adjacent to each other. The *minor* field should be located to the right; in this case, the product field is minor. Each progressively higher field (salesman, then branch) should be located to the left.

Automatic Punches

A number of 80-column machines punch serially from left to right, column by column. When this type of equipment is to be used in the processing, cards must be designed so that the data sequence does not conflict with the sequence in which results will be supplied and punched by the machine. Such a conflict can be avoided by studying the manufacturer's manuals of operation and the way processing is to be done.

Computers

Computer characteristics must be investigated to determine whether the card's design will affect processing speed. With many computers the ability to move and process blocks of data without rearrangement makes programming easier and increases processing speed.

Control Panel Wiring

Many 80-column card machine units are set up and controlled by means of wired *control* panels. The accuracy and speed of frequent wiring operations can be improved if the input card's data sequence is similar to the sequence of reading, punching, and printing.

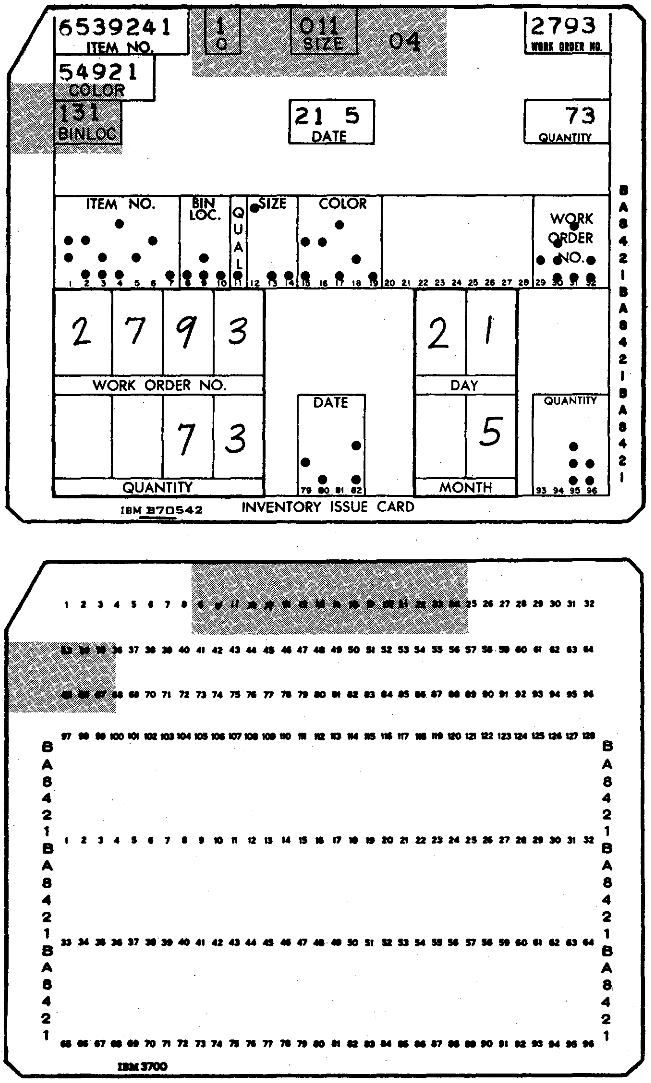


Figure 6.7 Dual and standard 96-column card in hopper of 5496 Data Recorder

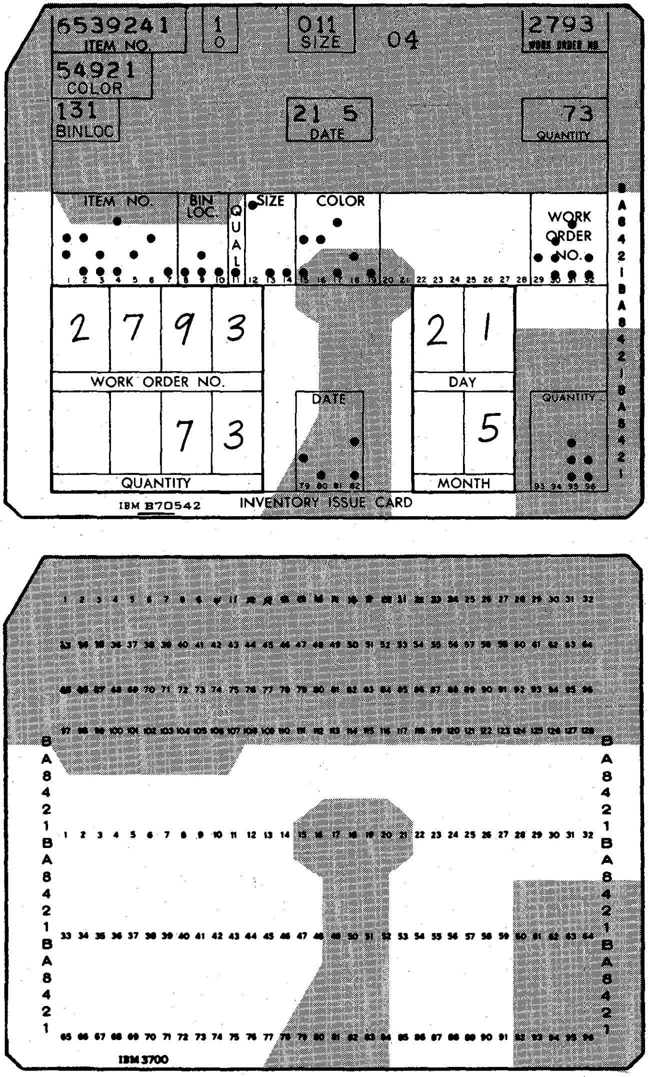


Figure 6.8 Dual and standard 96-column card in verify station of 5496 Data Recorder.

Manual Card Handling

If data are to be punched and printed on the 80-column card punch, information to be checked visually should be placed near the right or left end of the card. This makes it easier to fan and check a group of cards. If data are to be read from cards in a file, the reference data should be positioned so that it can be easily seen.

Positioning of data is somewhat more restricted on the 96-column card because of its smaller size.

96 - Column Card Layout Form

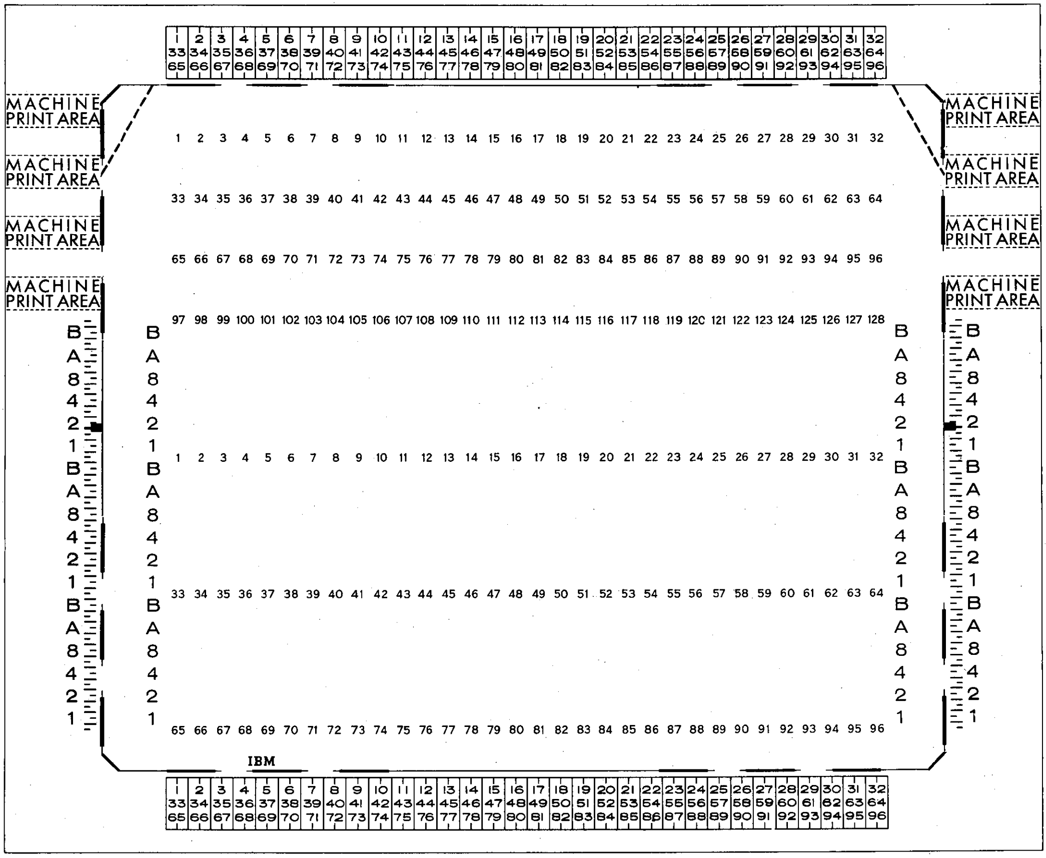


Figure 6.9 96-column card layout form.

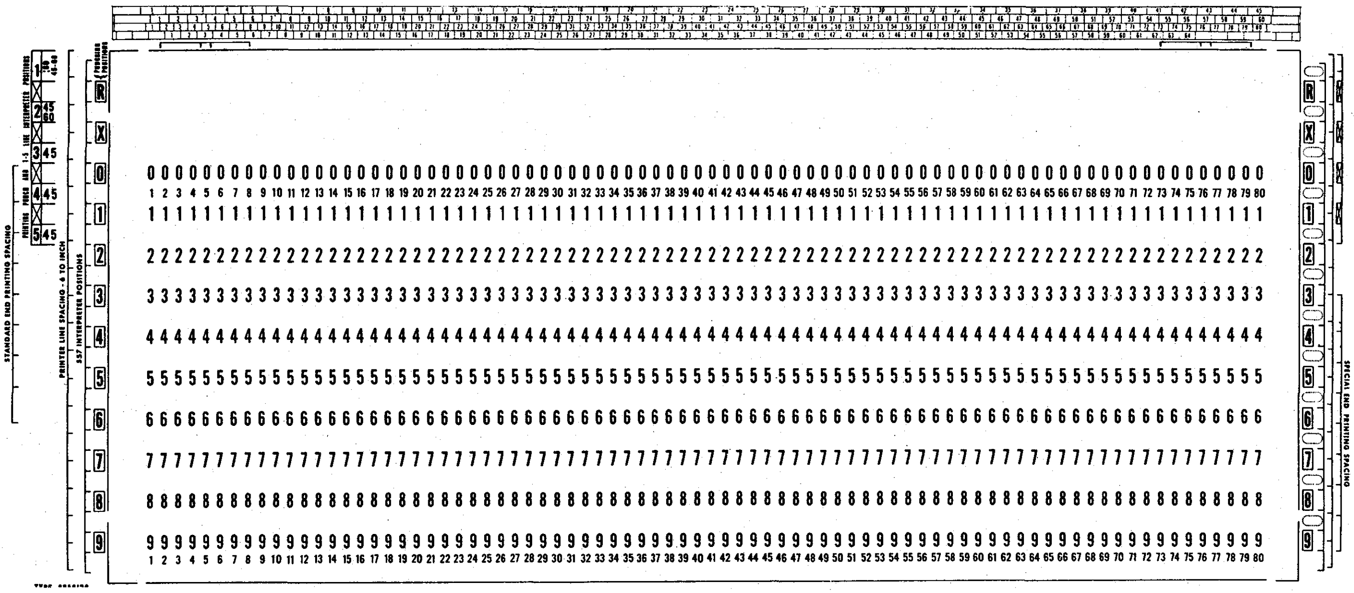

STANDARD END PRINTING SPACING
PRINTING PUNCH AND 1-5 LINE INTERPRETER POSITIONS
PRINTER LINE SPACING - 6 TO INCH
557 INTERPRETER POSITIONS
PROGRAM POSITIONS
SPECIAL END PRINTING SPACING
R
X
0
1
2
3
4
5
6
7
8
9

Figure 6.10 80-column card layout form.

Final Card Layout

Once field size and data sequence have been determined, the final card design can be laid out. Several types of forms are available for this purpose; samples are shown in Figure 6.9. To make design easier, these forms are considerably larger than standard size cards. By photography and reduction techniques, the forms can be reduced to proper size for making printing plates.

When drawing the card layout, there are a number of additional points to be considered. Again, the value of each must be determined by the card being designed and the application in which it will be used. Figure 6.10 shows the layout for a material accounting card. Much of the following discussion can be related to this form.

All field and box headings should be explicit and confusing abbreviations avoided. Headings for written information should be placed so that they force writing into the proper location. Make certain that adequate space is provided for this writing.

For dual 80-column cards, the right-hand side of each area or box containing handwritten data should be at least five card columns to the left of the field in which the information is to be punched. This prevents the data from being obscured when it is time to punch it.

The shield in the card punch that extends across the bottom of the bed between the punch dies and the reading station obscures about one-eighth inch of the card's lower edge. Avoid the use of this area for data that is to be read and punched into the card.

For easy reference, field headings should be located between the zero punching positions and the top of the card unless printing or mark sensing prevents it.

Keep titles and interpreted data between rows of punching so that they will not be obliterated.

Use pre-printed decimals and commas in boxes where amounts will be machine printed.

Colored cards and color strips may be used to identify types of cards. However, color should not interfere with the utility of the card, particularly the reading and writing of data.

Dotted lines should be drawn between columns (both card and mark sensing) to indicate the locations of decimal points and commas.

When designing a dual card, use one area of the card exclusively for handwritten data so that it will not be obliterated by punching. If there must be an overlap of handwritten and punched data, use the overlap for recording descriptive information that that can be easily reconstructed.

Whenever possible, retain the digits that show the location of the punching positions. This makes it easier to read punched information that has not been printed on the card.

Any information, handwritten or printed, that is to be visually checked should be placed near the right or left end of the card. In this location, the information can be more easily seen when the cards are fanned.

Mark sensing should be placed on the right end of the card so that the card can be held with the left hand and marked more conveniently.

Corner cuts can often be used to distinguish card types during clerical as well as machine processing operations. However, lower left cuts should not be used on some type machines because they may cause feeding problems. Be sure to check the manufacturer's recommendation.

Special consideration must be given to the design of short cards. Their position in the hopper of the processing equipment is predetermined and the card columns should correspond with the positions of the individual column sensing devices. This will make machine sorting and control panel wiring easier.

Questions and Exercises

This exercise involves the design of an 80-column dual card form to be used as a requisition for supplies from a small office stock room. Requisitioned items are stationery supplies: typewriter paper, scratch pads, pencils, paper clips, rubber bands, typewriter ribbons, and the like. Assume that the activity in the stock room has been surveyed and the following information established:

a. About 100 different items are carried in stock.
b. No more than 100 pieces of an item are issued on any one requisition.
c. A separate requisition is filled out for each different kind of item. For example, ribbons and paper cannot be filled from a single requisition.
d. Item description must include at least two fields: name (e.g., bond paper, rubber bands, paper clips) and unit of measure (e.g., box, dozen, sheets, pieces, bottle).
e. The stockroom is located in the same building with a number of offices and departments that obtain supplies there.
f. The stock room is tended by a clerk who services a window. Items are picked up by secretaries who carry the supplies back to their desks for their own use or for distribution to other office workers.

- g. Requisitions are originated by the secretaries and are authorized by managers of individual departments.
- h. At the end of convenient periods, filled requisitions are priced and extended (*quantity issued* X *unit cost*) to obtain the total cost of each item issued.
- i. Each department is charged with the cost of supplies used each month as an expense.

1. List the information fields that are written (or typed) on the requisition by the secretaries.
2. List the field or fields written by the managers.
3. List any fields written by the stock room clerk.
4. Which fields should be punched in the card?
5. Determine the number of columns needed to record each field by keypunching.
6. Design a proposed card to meet the requirements just listed.
7. Outline instructions to the secretary, stock room clerk, and manager for using the form.

With the instructor's guidance, the class should agree upon a single form that meets all estimated requirements. The operation of the stock room can be simulated as a class activity, with a number of requisitions produced as a result. If equipment is available, these may actually be keypunched, by class members or by a data processing department.

If 96-column equipment is available, the same exercise can be carried out using the smaller card as a requisition.

7 PROCEDURE CONTROL

Data processing procedures can be divided into three equally significant parts.

First: *the procedure must achieve some planned result* and, for commercial enterprises especially, produce documents that record those results. For example, a payroll procedure is designed to compensate employees for their work at periodic intervals according to agreed-upon rates of pay. The records produced include paychecks; earnings statements; payroll registers; and the many related reports for the employer and local, state, and federal governments.

An accounts receivable procedure is designed to keep an account of the money owed a business. Records produced include monthly statements of customers' accounts, notices of delinquent payments, credit records, and adjustments to account balances for payments and purchases.

Second: *the results must be accurate and dependable*. Requirements for accuracy differ with various types of procedures. In the case of payroll, where employee wages and tax records are concerned, there seldom can be any compromise with accuracy. It is in the employer's best interest to be sure that every employee on the payroll is paid correctly. In the case of accounts receivable, it is also in the best interest of a business to see that customer statements are accurate in every respect.

At the time a procedure is planned, it is necessary to assess the requirements for accuracy against the cost of obtaining it. For applications such as cost accounting, labor distribution, and inventory control, a degree of accuracy somewhat less than absolute can usually be tolerated.

To illustrate this point, there is a story of a munitions plant during World War II that received an emergency order for a number of specially designed small arms. Because of the critical and unusual nature of the order, the plant manager assigned his most dependable expediter to take personal charge of production. True to his reputation, the expediter was able to effect a small miracle. By largely ignoring established operating procedures, he was able to have the complete order ready ahead of schedule.

As the plant manager and his man stood on the shipping platform to watch the last of the crated pieces being loaded on a truck, the manager was extremely generous with praise for a job well done.

As they turned to leave, the manager paused a moment, then asked, "By the way Charlie, how much did those things cost?"

Said Charlie, in a classic reply, "About half as much as they will if we try to find out!"

It should also be noted that a data processing department is nearly always a service unit within a larger organization. The department accepts information in some form to be processed. Usually, the incoming information is in the form of source records which must be transcribed to punched cards for further processing. The processing steps may include sorting, classifying, reproducing, calculating, report printing, summarizing, and so on. Processing may also include the preparation of records for further processing by other types of equipment, including computers and communication equipment.

No matter how complicated the procedure or what the end result will be, it must be remembered that the machine processing department does not *originate* any information. The department can only handle data as received.

It follows, therefore, that final results can be only as accurate as the input data. Wrong data can often be processed as easily and quickly as that which is correct. Systems engineers have a rule that applies to this situation:

"Garbage in; garbage out!"

In general, the requirements for accuracy in a procedure may be summarized as follows:

1. The financial status of the business must be accurately reported to management and the stockholders.

2. Customer, employee, and vendor records must be maintained with a minimum of error.

3. Accounting for tax purposes, company earnings statements, and government reports must be accurate within the requirements of law.

4. The accuracy of operating reports must conform to the requirements imposed by internal and external accounting practices. Procedures must be modified to meet these requirements.

Third: the *control* of a *procedure must be exercised* in such a way that the *results are achieved* at *reasonable cost* and within a *specified time limit*. This might also be called the *management* of a procedure.

For example, a payroll procedure must be managed so that employees are not only paid correctly and according to law, but also so that they are paid *on time*. In accounts receivable applications, customers must be billed by certain dates and their account balances must be kept as current as possible.

Data must be handled in a fashion that makes processing as efficient as possible. This may involve such steps as machine duplicating of card records so that two different operations can be carried out at the same time.

Control must provide for the organization of a procedure in such a way that, in the event of error, the trouble can be found and fixed with a minimum of lost time. For example, it is possible for a card record to be jammed or torn by the equipment or by careless handling. If this happens, the corresponding source record must be available for reference so that the damaged card can be repunched and put back into the file without costly interruption.

Stringent control must be exercised to prevent theft or fraud that may affect the financial standing or reputation of a business. This requirement particularly applies to the production of records that can be negotiated for cash or other value; for example, checks of all kinds, bonds, and stock certificates.

The procedure must be designed to leave an easily followed *audit trail*, that is, a means of auditing by tracing the flow of data from input to final output.

This chapter is concerned with the methods used to establish control and to assure accuracy over the transcription from source record to punched-card record. Direct machine verification is discussed in some detail with descriptions of the equipment used in this operation.

As other card processing operations are presented in later chapters, appropriate methods of control will also be described.

Record Count

The use of a record count is one of the simplest ways to maintain control over the movement of documents from one stage of processing to another. An original count is obtained from the tally of the number of records in a given file or set. The count can be useful as a quick check that all source documents have been received into a data processing department.

Record count totals should be established at the location where the documents are originated or first assembled. The total is obtained by a manual count. The count is then adjusted whenever any records are added or withdrawn from the file. If documents are processed or transferred to a new location, the records are recounted and the result compared with the original or adjusted total. If the quantity agrees with the control total, it is accepted as proof that all records have been transferred, or processed in some other manner as called for by the procedure.

Record counts are usually established by batch. This is desirable when all records involved in a given procedure may not be received into the department at one time. After the batch totals are set up, they are added together to form the single total for all records processed.

Although the record count is useful as proof that all records of a given set have been accounted for, it is difficult to determine the cause of error if the record count is out of balance. A failure to balance against a control total does not help to locate a missing record nor does it indicate which record may have been processed more than once. Therefore, some provision must be made to check the error by using a duplicate set or a listing known to contain the proper number of records.

An error in a record count is merely the first indication that something is wrong in the operation of the procedure. More elaborate controls must be established to locate the cause of the error.

Control Total

Whenever the term "control total" is used, it is normally taken to mean a total of some amount or quantity fields in a group of records. However, as just explained in the section describing record counts, the term can be applied to almost any accumulation of factors or items that might be used for procedure control.

The control total is accumulated either manually or by machine when a record set is first originated or assembled. Totals are usually taken by small, easily handled groups of records to form batch totals which can be added together later to form the *grand* total.

For example, assume that a batch of invoices is received into the machine data processing department. Attached to the batch is a transmittal slip stating that there are 169 records in the batch. Now assume that these records are to be keypunched.

The first step in processing is to verify that all records have been received. This is done by fanning through the bundle and manually counting the documents. If the count proves to be 169, then it may be assumed that all records have been received.

Next, assume that it is the job of the receiving clerk in the keypunch room to establish a total of the amounts of the invoices for control purposes. The clerk does this by preparing an adding machine tape of the *totals* from each invoice. The tape is attached to the bundle of invoices and the total is entered on the transmittal slip.

The invoice data is now keypunched into cards in the normal manner, with one card prepared for each item. But the totals of each invoice are *not* punched—only amounts for the individual items.

The cards are machine verified for identification data, quantity, and so on. Item totals are *not* verified. When all cards for the batch have been completed, the bundle of invoices along with cards, transmittal slip, and amount total are sent to an accounting machine. An operator feeds the punched cards into the machine to add the *punched* item amounts. At the same time the card data is listed on a sheet of paper showing all the information punched. The machine is controlled to print totals by invoice number so that the listing is a composite record of all information transcribed. At the end of the listing the machine also prints a grand total.

If all the previous work has been done correctly, the grand total from the accounting machine should balance to the original adding machine tape.

If there is an error, each machine total can be checked against the corresponding original invoice total. When the error is located, each detail item is checked until the exact difference is accounted for. It may be that an error was made in transcribing more than one item. It is also possible that an error was made in taking the adding machine total.

When all corrections have been made and the two totals reconciled, a final and correct listing is run on the accounting machine. This listing can now serve all control purposes that the original documents served. Each time the corresponding card records are processed, they can be balanced against the control listing. In case of error, all source information is readily available for checking.

Unit Count

Closely related to the record count, the unit count differs because no count is established for the actual number of records or cards. Instead, the total is set up for the number of different *units* in a file. In some applications, particularly payroll, units may mean employees.

For example, a payroll is always made up for a known number of employees. Therefore, if a batch of weekly clock cards are received in the machine room as part of a payroll procedure, there should be one card for each employee. This total is established first by a record count and may be verified by the personnel department or by the various department managers or timekeepers.

As the payroll is processed, employee clock cards may be combined with other cards, such as master name cards with employee name, serial number, social security number, rate of pay, and so on. Tax records and previous year-to-date earnings cards may be included with other deduction cards. When the final payroll is ready for processing, there is an undetermined number of cards in the file for each employee. However, the *total number* of employees must still balance to the original unit count.

When the final payroll records are prepared, the card processing equipment can be controlled to count the number of minor totals taken on the report—this is one for each employee. Regardless of the number of cards in the file, the machine counts totals only and consequently, the number of employees. This total is carried through each step of the payroll procedure. Ultimately, when checks are written, the unit count control total must equal the number of employees on the payroll.

The unit count is often combined with other control totals to provide procedure accuracy. In combination, the two totals can be used to predetermine amounts or quantities on any reports produced in a given procedure.

For example, when preparing to process a payroll, the total number of hours worked by all employees can be established from the original clock or job cards. This figure then becomes the control total of payroll hours for all subsequent reports. Totals may be broken down by group or department to make it easier to localize any possible errors. The sum of all totals must balance back to the complete original control total.

Control totals are usually set up by batches of convenient size, such as department, location, office, account, or division. By this method, each group of records may be balanced as it is processed. Corrective action, when needed, can be limited to small easily checked groups rather than one grand total.

To again use a payroll procedure for illustration, it is usually best to set up control totals by department, location, or office. As the various reports are run, each group can be balanced to these basic control totals. Control of earnings, tax amounts, and deductions for a current period, and cumulative year-to-date amounts will be discussed with the operation of calculating equipment and accounting machines (Chapters 11 and 14).

Self-Checking Number

The dependable operation of almost any data processing procedure depends to a considerable extent upon the accuracy of control numbers. These are the identification codes assigned to records, items, persons, locations, accounts, and so on, to make efficient machine accounting possible.

The entry of an incorrect control number into a procedure or processing cycle always produces an error of misplaced data. That is, the information relating to the control number, usually values or quantites, is improperly identified. Consequently, that information may be lost. Or, the use of a wrong number can associate corresponding information with another identification code which by coincidence happens to match the error number.

The self-checking number is used with numerical identification fields to assure accuracy whenever the field is written or transcribed. The number assures a very high degree of accuracy for those fields, even on the original source document. When used in punched card fields, special features must be installed on the equipment by the manufacturer.

One typical use of the self-checking number is to positively identify the recording of items in an inventory. In such an application, handwritten original documents are common. And in nearly all cases, writing the proper part number on a record is just as important as putting down the correct quantity of parts moving in or out of a stock room. In this application particularly, an error in identifying the item or part can multiply into a surprising number of other errors later on in the procedure.

For example, suppose a stock room attendant releases a quantity of item 1234 but inadvertently records the withdrawal of item 2134. It is assumed that 2134 is also a valid part number for another item in stock.

When the periodic inventory report is run, it may show that item 2134 is nearly out of stock and should be reordered. Consequently, a surplus of the wrong item eventually gets to stock. Meanwhile, item 1234 shows a balance on the report of more pieces on hand than are actually on the shelf. If item 2134 is stocked at a cost of $100 each and item 1234 at a cost of only $10 each, then the error has caused an unnecessary and perhaps serious inflation of inventory costs.

Other costly errors can accumulate as a result of the incorrect recording of the part number. Suppose that part 2134 was recorded as withdrawn from stock in a maunfacturing plant where the cost of parts is charged against a shop order for work being done for a customer. In this case, the cost of the order is increased by $90 for each part used. Remember, the correct parts were *issued*; only the identification number was wrong. Conversely, if the cost of the recorded parts is less than the cost of the ones actually used, a serious loss could occur for the plant.

The use of the self-checking number requires that a *check digit* be developed for each basic code number to be self-checked. Once the check digit for the basic code number has been determined, it is added to the units or last position of the number. The check digit can be developed by a self-checking number generator on the keypunch, by computer calculations, or by manual arithmetic. Thus the self-checking number is one digit longer than the original base number.

A number of terms should be defined before describing examples of two different check digit systems.

Transcription error: This is an error of substituting one or more digits in a number for the correct digits.

Examples: 0 for a 1, 2, 3, 4, . . .
1 for a 0, 2, 3, 4, . . .
2 for a 0, 1, 3, 4, . . .

Transposition error: This is an interchange of two digits in adjacent columns.

Examples: 457843 for 458743
865745 for 865754

Double transposition: This is the interchange of digits in alternate columns.

Examples: 457843 for 454873
865745 for 875645

Random error: A combination of two or more of the above or any error not covered by the above.

Weights: The amount by which each digit in the control number is multiplied to arrive at a product.

Example:	4	5	7	8	4	2	=	number
	7	6	5	4	3	2	=	weights
	28	+30	+35	+32	+12	+4	=	141 = sum

Modulus: The number used to divide the sum of the digits in the control number to arrive at a remainder.

Examples: *141* divided by *10* = *14* + *1* remainder
141 divided by *9* = *15* + *6* remainder
141 divided by *11* = *12* + *9* remainder

Check digit: The number added to the sum of the products (weights X control number) to produce a number which when divided by the modulus will give a remainder of zero. Or, it is the difference between the sum of the products and the next highest even multiple of the modulus.

Examples: *29* divided by *10* (modulus) = *2* + *9* remainder. *9* + *1* = *10* or *30* − *29* = *1*. Therefore, *1* is the check digit.

141 divided by *11* (modulus) = *12* + *9* remainder. *9* + *2* = *11* or *143* − *141* = *2*. Therefore, *2* is the check digit.

Modulus 10

Modulus 10 is designed to detect either incorrect keying of a single digit or a single transposition. The following arithmetic process is used to generate the check digit, regardless of whether it is generated by a keypunch, a computer, or manually. Refer to Figure 7.1.

Example:

Basic code number		6	1	2	4	8		
Units and every alternate position of basic code number		6		2		8		
Multiply by 2						X2		
Product	1	2		5		6		
Digits not multiplied by 2			1		4			
Cross add		1 + 2 + 1 + 5 + 4 + 6						= 19
Next higher number ending in zero								20
Subtract crossfooted total								−19
Check digit								1
Self-checking number		6	1	2	4	8	1	

Other examples:

	Self-checking number	
Basic code number	Basic code	Check digit
45626	45626	9
30759	30759	5
73074	73074	7

Figure 7.1 Self-checking number. Modulus 10.

1. The units position and every alternate position of the basic code number is multiplied by 2.

2. The digits in the product and the digits in the basic code number that are *not* multiplied by 2 are cross added.

3. The cross-added total is subtracted from the next higher number ending in zero.

4. The difference is the check digit.

In this system, the space and zero have the same numeric value. Therefore, spaces can be substituted for insignificant zeros.

Once the number is generated, no special knowledge is required to write the numbers on an original document or to punch the numbers into cards. With an 80-column keypunch, the operator keys and punches the number, including the check digit, as it appears on the source document. The accuracy of the keying and the validity of the self-checking number can be automatically verified by the machine.

When the number on the source document is correct and the number keyed correctly, the punching operation is not interrupted. More than one self-checking field may be checked in one card, but the fields must be separated by at least one card column. If no errors are detected after punching all the self-checking fields in a card, an 11 hole is automatically punched after column 80 (in what would be column 81).

If the number is incorrectly recorded on the source document, or if the number is punched incorrectly, an error is indicated after the last digit of the self-checking field is punched. A red light appears on the keyboard, a 12 hole is automatically punched in the units position (check digit) column, and the keyboard locks. When an error occurs, the automatic 11 punch in column 81 is suppressed.

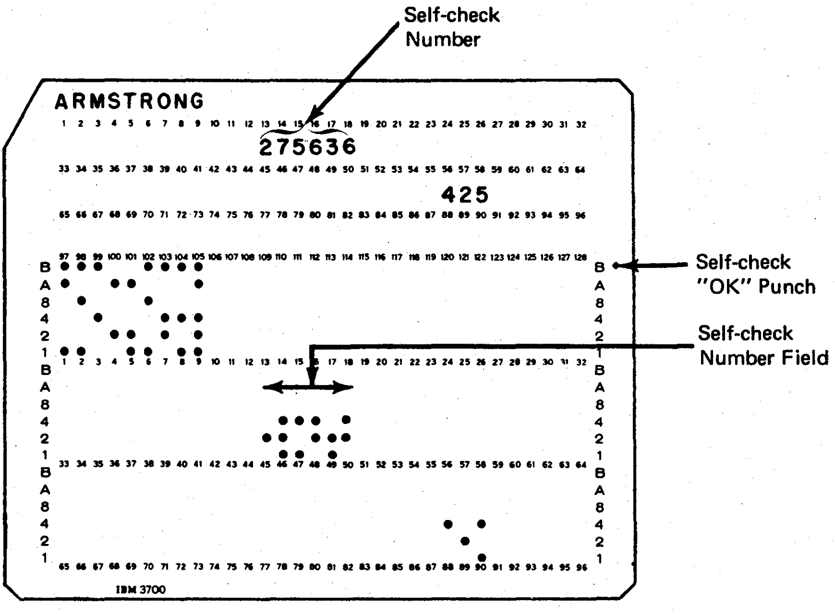


Figure 7.2 Self-check punch location. 96-column card.

The 5496 Data Recorder does not generate the self-checking number; self-checking fields are verified as they are punched. When no errors are detected, an OK punch is automatically made in the margin of the card near column 32. Figure 7.2 shows the 96-column card with a self-checking number field that has been correctly punched.

With the modulus 10 system, the earlier example of the incorrectly recorded part number can be used to prove that a self-checking number would have revealed the error. First, the check digits for the two numbers are generated as follows:

Basic code numbers	1 2 3 4	2 1 3 4
Units and alternate positions of basic code numbers	2 4	1 4
Multiply by 2	X2	X2
Product	4 8	2 8
Digits not multiplied by 2	1 3	2 3
Cross-add	1+4+3+8 = 16	2+2+3+8 = 15
Next higher number ending in zero	20	20
Subtract cross-added total	16	15
Check digit	4	5
Complete number	12344	21345

Now, the stock room attendant must write a *five*-digit part number for each item issued. If the exact same transposition is made using this system, the attendant would write the part number 12344 incorrect as 21344. Because it has the wrong check digit, this number would be caught as an error on the keypunch.

Modulus 11

Modulus 11 is designed to detect single digit mispunches, single transpositions, and double transpositions. The main feature of this system, distinguishing it from other self-checking number systems, is that it is based on a weighted checking factor for each digit in the basic number being tested. Regardless of how the self-checking digit is generated, the following arithmetic process is used. Refer to Figure 7.3.

Basic number: 943457842

Write digits of basic number: 9 4 3 4 5 7 8 4 2

From right to left, write checking factors: 4 3 2 7 6 5 4 3 2

Add the products: 36 + 12 + 6 + 28 + 30 + 35 + 32 + 12 + 4 =
Total 195

Divide: 195 divided by 11 = 17 plus a remainder of 8

Subtract: 8 subtracted from 11 = 3 (the check digit)

The new self-checking number is: 9434578423

Figure 7.3 Self-checking number. Modulus 11.

1. Each digit position of any basic number is assigned a weight or checking factor. These factors are: 2,3,4,5,6,7,2,3,4,5 . . ., starting with the units position of the number and working toward the high-order digit.

2. In the example shown in Figure 7.3, write the number as illustrated, leaving space between the digits.

3. Below each digit, starting at the right and working to the left, place the corresponding weighting factor.

4. Multiply each digit by its weighting factor and add the products.

5. Because this is a modulus 11 operation, divide the sum of the products by 11, and subtract the remainder from 11. The result is the check digit.

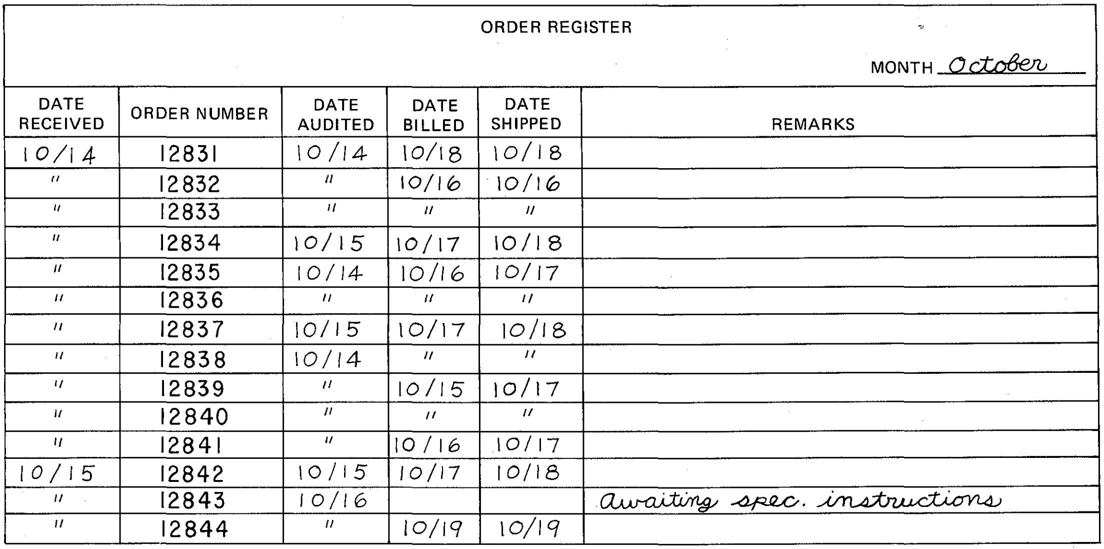
In the modulus 11 system, basic numbers that require a check digit of 10 cannot be used as self-checking numbers. The coding system must be adjusted to eliminate such numbers that are to be self-checked. If an operator is generating check digits on an 80-column keypunch, and punches a basic number requiring a check digit of 10, the machine indicates an error. A 12 punch is placed over the check digit column, the error light goes on, and the keyboard locks. The operator must release the card and substitute another basic number.

When the calculations on the basic number result in a check digit of 11, the digit 0 can be substituted and appended to the basic number to make a valid self-checking number. Eleven and zero have the same value in this part of the calculation.

Document Register

A *document register* can often be used effectively to control the flow of records in and out of a data processing center. Such a register is most effective where single records or small batches are received at frequent intervals by a control clerk. The register is maintained as proof of receipt, processing, and eventual transfer of the documents from the center to another area. Further processing of some sort may or may not be done there. Figure 7.4 shows a sample order register as it might be used in a billing and accounts receivable procedure.

To begin the system, all the orders are serially numbered at the point of origin. Each order must have a number, and orders



ORDER REGISTER

MONTH October

DATE RECEIVED	ORDER NUMBER	DATE AUDITED	DATE BILLED	DATE SHIPPED	REMARKS
10/14	12831	10/14	10/18	10/18	
"	12832	"	10/16	10/16	
"	12833	"	"	"	
"	12834	10/15	10/17	10/18	
"	12835	10/14	10/16	10/17	
"	12836	"	"	"	
"	12837	10/15	10/17	10/18	
"	12838	10/14	"	"	
"	12839	"	10/15	10/17	
"	12840	"	"	"	
"	12841	"	10/16	10/17	
10/15	12842	10/15	10/17	10/18	
"	12843	10/16			awaiting spec. instructions
"	12844	"	10/19	10/19	

Figure 7.4 Order register.

are transmitted to the data processing center in serial number sequence whenever possible, although there may be some exceptions. As orders are received by the control clerk, each one is checked off against a preprinted corresponding serial number on the order register. Missing orders are immediately identified by this posting operation. The serial numbers can also be copied by hand if this is more expedient.

Columns are provided on the register to record the date when each order is received and to check the completion of other operations as well. Processing points may be either checked off or recorded by entering the date in the proper column. In this way, the register serves not only to control actual processing, but also to provide the time each major operation is completed.

Next, it is assumed that the orders are to be transcribed to punched cards and then verified. Both operations are checked off in the proper columns. From the punched cards, it is then assumed that an invoice is prepared. A number of detailed steps can occur here, all carried out within the machine room. When the invoice is completed, it is sent to the shipping department for entry of the date shipped. A column is provided for remarks.

The register can be used to flag delays in processing as shown in the case of number 12843 (Figure 7.4). After several days, this order has not been shipped.

It should be noted that a rather elaborate control of this kind would be maintained only for important and valuable records. In this case, the register shows the response time of the procedure, and probably the business, from receipt of any order to its final shipment.

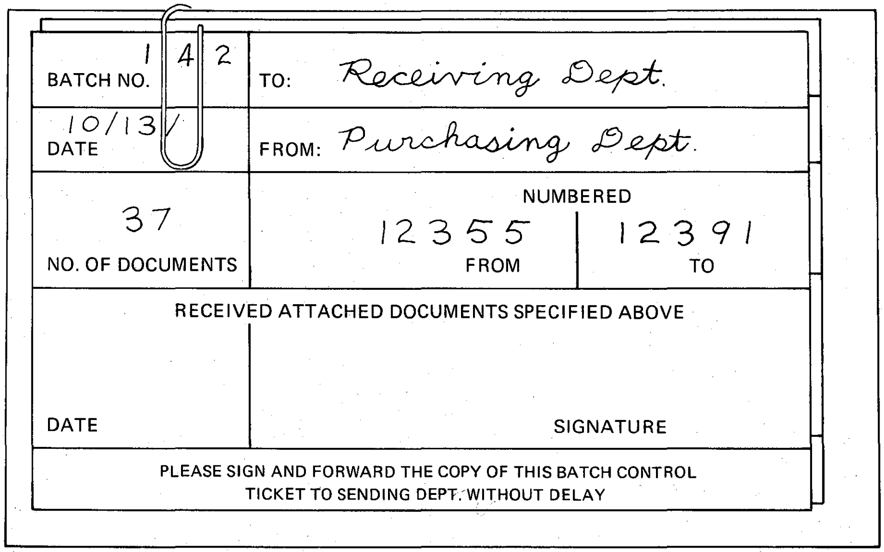
If the volume of items on the register is large enough, it is also possible to mechanize this document by preparing a dual punched card for each line, keypunching the cards, and listing the register on an accounting machine.

Serial Numbers for Batch Control

Serial numbers on documents can be used to check that all proper records are included in a procedure. By arranging the records in serial number sequence, a check can be made for missing or duplicate documents at any point in the processing. This system furnishes effective control of records such as checks, drafts, bonds, and other valuable papers when all such documents must be accounted for throughout a processing cycle.

If the document is a punched card, the serial number can be both printed on the card and punched as holes. Cards can be sorted by serial number and counted automatically. Missing or duplicate numbers can also be located by machine.

Serial numbers can also be used to identify batches of records. The number of documents in each batch is recorded on a control



1 4 2 BATCH NO.	TO: Receiving Dept.	
10/13/ DATE	FROM: Purchasing Dept.	
37 NO. OF DOCUMENTS	NUMBERED 12355 FROM	12391 TO

RECEIVED ATTACHED DOCUMENTS SPECIFIED ABOVE

DATE	SIGNATURE

PLEASE SIGN AND FORWARD THE COPY OF THIS BATCH CONTROL
TICKET TO SENDING DEPT. WITHOUT DELAY

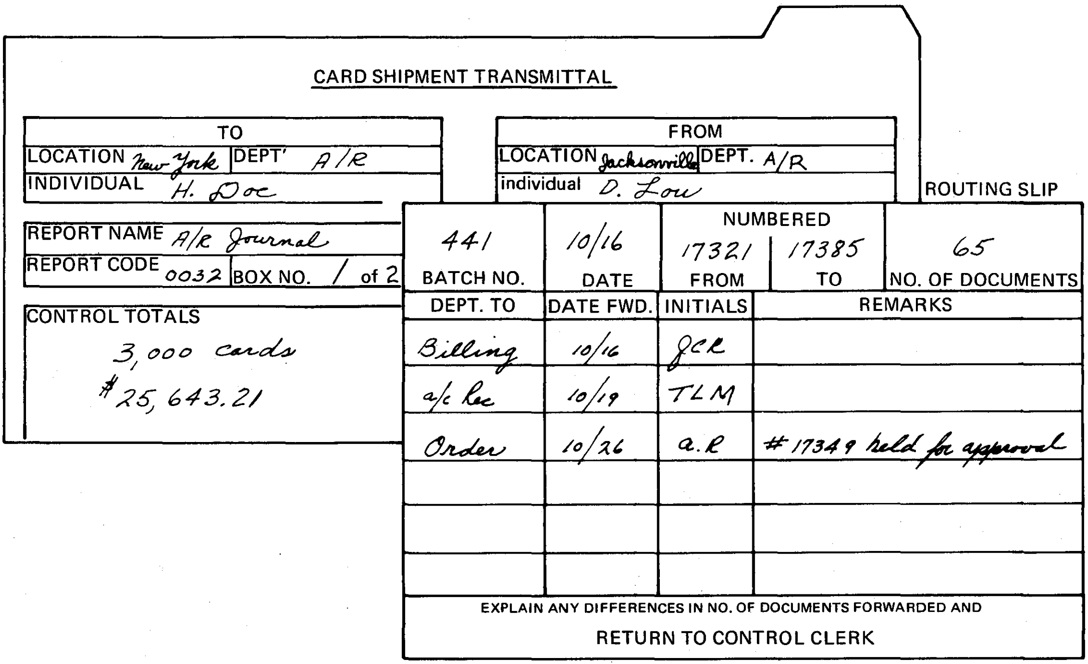
Figure 7.5 Batch control ticket.

slip or ticket. In addition, the first and last serial numbers in the batch are written on the control ticket. The size of a batch should always be kept to an easily handled number of records.

Figure 7.5 is an example of a batch control ticket. Notice that space is provided to record the sending and receiving departments, batch number, date, and the first and last serial numbers in the batch. With this record, both the sending and receiving departments can account for all records.

Transmittal and Route Slips

A letter of transmittal describing a group or batch of documents is frequently used to set up control and transfer responsibility when documents move from one department or location to another. The transmittal is usually a printed form with spaces to indicate the variable information for the batch. Refer to Figure 7.6.



CARD SHIPMENT TRANSMITTAL

TO	
LOCATION New York	DEPT' A/R
INDIVIDUAL H. Doe	

FROM	
LOCATION Jacksonville	DEPT. A/R
individual D. Lou	

REPORT NAME A/R Journal

REPORT CODE 0032 BOX NO. 1 of 2

CONTROL TOTALS

3,000 cards

#25,643.21

ROUTING SLIP

		NUMBERED		
441	10/16	17321	17385	65
BATCH NO.	DATE	FROM	TO	NO. OF DOCUMENTS
DEPT. TO	DATE FWD.	INITIALS	REMARKS	
Billing	10/16	JCR		
a/c Rec	10/19	TLM		
Order	10/26	a.R	#17349 held for approval	

EXPLAIN ANY DIFFERENCES IN NO. OF DOCUMENTS FORWARDED AND

RETURN TO CONTROL CLERK

Figure 7.6 Transmittal and route slips.

When the volume of work or the number of people who may perform any given operation is large, it may be desirable to fix responsibility and account for documents passed from each operation to the next as well as from one department to another. In this case, a route slip is used, either in addition to or in combination with the letter of transmittal. The route slip is similar to the batch control ticket. However, each department or operational step which the accompanying documents pass through is identified together with an indication of the processing time and the operator or clerk responsible for each job. Responsibility is fixed and the means to effect a degree of work control as well as document control has been incorporated into the same form.

Questions and Exercises

1. Explain how the planned results of procedures must sometimes be modified to meet requirements for accuracy and control.
2. What is an audit trail?
3. To what extent is the record count useful in establishing procedure control?
4. Show how a record or unit count can be combined with control totals to provide effective control.
5. What is the essential difference between the modulus 10 and modulus 11 self-checking numbers?
6. List the purpose of each of the following: document register, serial number, transmittal and route slips.
7. Review your procedure for use of the stock room requisition as proposed in the exercise following chapter 6. Decide what controls, if any, should be established over the origination of this form.
8. Present your controls for class discussion.

SORTING AND CLASSIFICATION 8

Sorting is the process of ordering and arranging a set of records in some given sequence. In punched card data processing procedures, the sequence is usually numerical; that is, arranged in the same order in which we count. For example, in counting from 1 to 100, the count *ascends* from 1 (the lowest order and first number in the series) to 100 (the highest order and last number).

Ascending numerical sequence is the most common filing order for records that have numerical identification or control fields, such as part number, social security number, and account or customer number. Numerical sequence has an advantage over other filing orders because it is most easily handled by punched-card sorters in the least number of passes of cards through the machine. Numerical sequence also has historical preference with punched-card equipment since early models of these machines could process only numerical data; alphabetic or other special characters could not be handled.

Reverse, or descending numerical sequence is sometimes desirable. In the series 1 through 100, 100 would then be the first record in the set; 1, the last. Such a sequence might be useful where records are filed by date. If day, month, and year are numerically represented (for example, July 4, 1970 as 07 04 70), the first record in the file would be the one with the most recent date. The last record would be the one with the oldest date. This sequence has obvious advantages where records are filed in order of their age.

Variations on the numerical sequence can also be used. Although more difficult to manipulate by machine, such sequences can nevertheless be established where it is necessary to group records by categories or codes that may not correspond to a straight ascending or descending sequence. One such requirement might be the arrangement of records in special groups or account numbers where all records of the series 1,000, 4,000, and 8,000 might represent charges to raw material; all records of the series 2,000, 3,000, and 5,000 might represent charges to finished goods; and all records of the series 6,000, 7,000, and 9,000 would be charges to inventory expense.

Alphabetic sequences are constructed in the same manner as numerical sequences. The basic order of a normal ascending series begins with the letter A as the first or *low*-order character and ends with the letter Z as the last or *high*-order character. Reverse alphabetic arrangements are possible though unusual. Arrangements in special or variable sequences may be fairly common in name files of various kinds but not in fields designed for machine processing.

Alphabetic sorts on punched-card sorters require more passes for a given alphabetic field than for a numerical field with the same number of characters. Hence the inherent advantage of simplified handling is an incentive to maintain numerical sequences whenever possible.

Many record fields also contain the special characters of punctuation and report printing: commas, periods, ampersands, parentheses, number signs, and the like. These characters can also be sorted into a predetermined sequence by both punched-card machines and computers. This is necessary because special characters often appear in the same fields with both alphabetic and numerical information. For example, nearly all address fields contain name, street and number, city, and state: John Doe, 45 Pleasant Lane, County Center, California. The ascending sequence of special characters with relation to digits is set as a standard convention by manufacturers throughout the office equipment industry.

Sets of records can also be arranged in sequence by one or more fields. For example, a file of insurance records could be arranged by policy number as the first or *minor* sequence in the set. The records could then be sorted by issuing office number as the next or *intermediate* sequence, then by class of insurance as the last or *major* sequence. The resulting order of the file is the grouping of records by class of insurance, by office number within each class, and by policy number issued by each office.

Punched-card accounting machines can be controlled to recognize at least three orders or sequences within a given set of records. Subtotals of amount fields can be accumulated and printed within each minor, intermediate, or major group of records as the cards are fed to the machine.

Machine Operation

Figure 8.1 shows one type of IBM sorting machine designed to operate with 80-column cards. An explanation of how this machine works will illustrate the basic principles of operation for many other types of punched-card equipment.

The file or deck of punched cards to be sorted is placed in a hopper at the right end of the machine, facedown, 9-edge toward the machine. Two types of hoppers are available with IBM sorters: the standard vertical hopper and a file-feeding device (Figure 8.2). The standard hopper holds about 1,200 cards, the file-feeding device about 3,600 cards.

Cards are placed in the machine by an operator, usually from a card file or stacking tray. Depending upon whether or not the

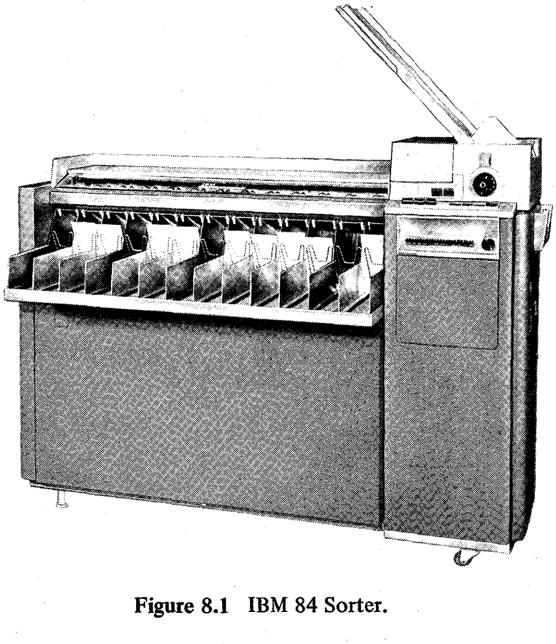


Figure 8.1 IBM 84 Sorter.

sequence of the cards before sorting is to be maintained, the operator normally fills the sorter from the front of the file or tray.

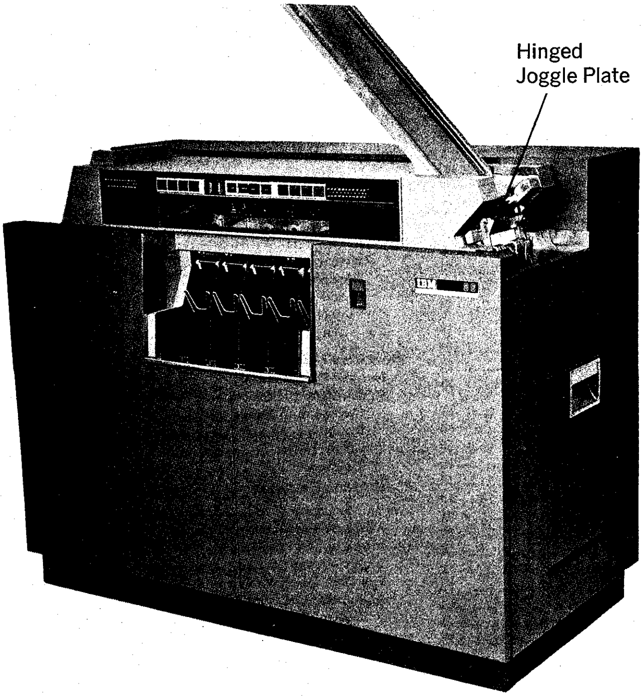


Figure 8.2 IBM file feeding device.

When the machine is started, cards are mechanically fed into the transport mechanism, one at a time, from the bottom of the deck. Cards move horizontally across the machine, from the operator's right to left, into stackers or *pockets.* Each card passes between a metal roller and a small wire brush or between a small light bulb and a light-sensitive diode, depending upon the machine model. The metal bristles of the brush are shaped to fit exactly through the rectangular holes in one column of the card. The sorter reads only one card column at a time.

The brush (or the light-sensing device) is movable and can be placed over any of the 80 columns by the operator. When there is a hole in the column being sorted, the brush drops through the card and touches the metal roller for an instant (Figure 8.3). (With the photoelectric sensing method, light shines through the hole onto the diode.) The metal contact (or the light sensing) creates an electrical impulse which controls the mechanism of the machine. Furthermore, the impulse occurs at a time that exactly corresponds to the position of the hole with relation to the feeding cycle. Thus, if a hole is sensed in the 4 row, that impulse will occur at "4 time" in the feeding cycle and is therefore read by the machine as a 4 (Figure 8.4). When a hole is sensed in the 1 row, the card is at the 1 position in the cycle. The machine consequently reads that hole correctly as a 1. The card is then automatically transported to a pocket or stacker by a *chute blade* that corresponds to the position of the punched hole in that column.

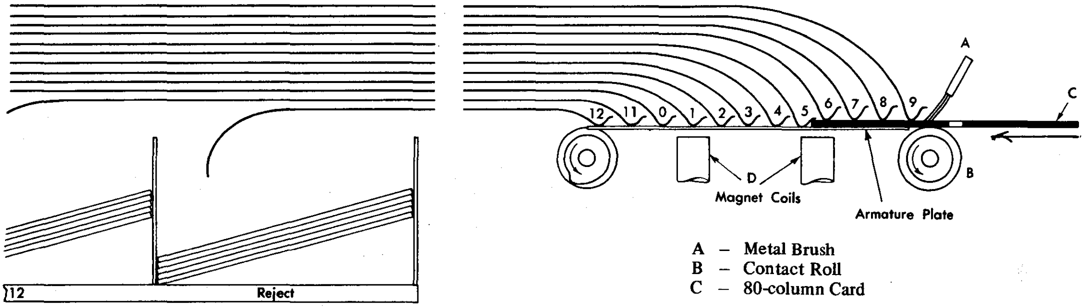


Figure 8.3 4-punch in 80-column card entering sorter selection device.

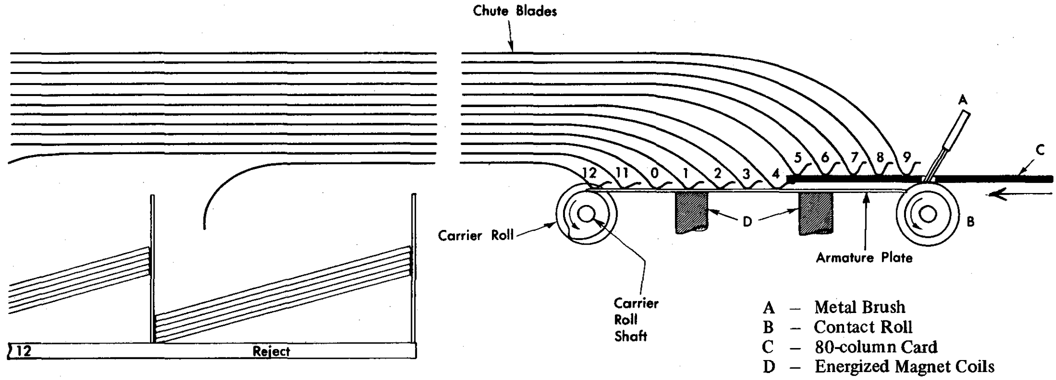


Figure 8.4 4-punch sensing directs card to 4 pocket.

There are 12 chute blades or bands of metal, varying in length, that guide the cards from the sort brush to the pockets

(Figures 8.3 and 8.4). Each chute blade extends from a point immediately past the sort brush to the opening above one of the numbered pockets.

Since there are 12 possible punching positions in any one column, there are also 12 pockets in the sorter plus one pocket for cards that have no holes punched in the column being sorted. For example, if there is a hole in the 3 position of the column, that particular card is directed to the 3 pocket. If there is a hole in the 1 position, the card goes into the 1 pocket. If there is no hole, the card is directed to the pocket reserved for blanks, the *reject* pocket.

For 80-column cards, IBM offers sorters in three models, each operating at different speeds:

Sorter	*Cards per minute*
Model 82	650
Model 83	1,000
Model 84	2,000

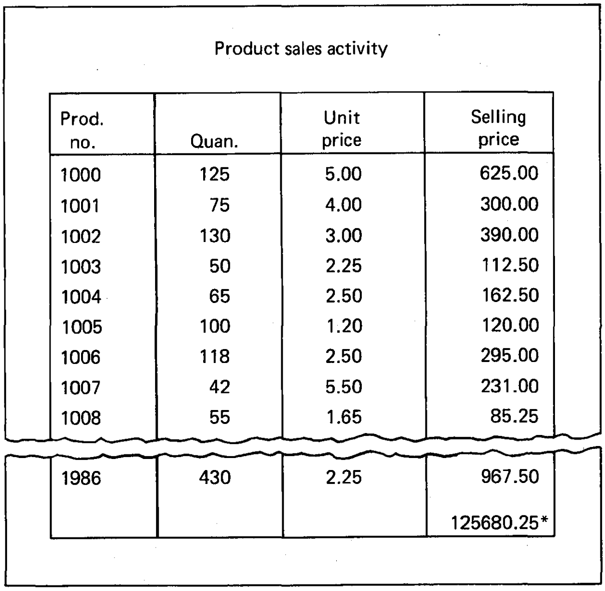
Sorting for Printed Reports

Printed output is often the end result of a machine accounting procedure. Invariably, the information shown is arranged in some organized form and in a required sequence. In many cases, information is ordered in a certain sequence for one report, then reordered in another sequence for a second report. The same card records may be used for a number of reports. It is only necessary to sort them properly beforehand.

For example, assume that a business requires reports to assist management in guiding sales policy. One file of punched-card sales transactions can furnish all the necessary information when arranged as follows:

1. *Product Number.* All sales of each product appear together in product number sequence (Figure 8.5). The report shows quantity, unit price, and total sales value for each item. If other descriptive information is available in the cards, sales trends can also be shown by color, style, model, size, and so on. To furnish these analyses, the cards are sorted by the corresponding identification or descriptive field.

2. *Customer Number.* On this report (Figure 8.6) all sales for a period are tabulated by individual customer. Additional in-

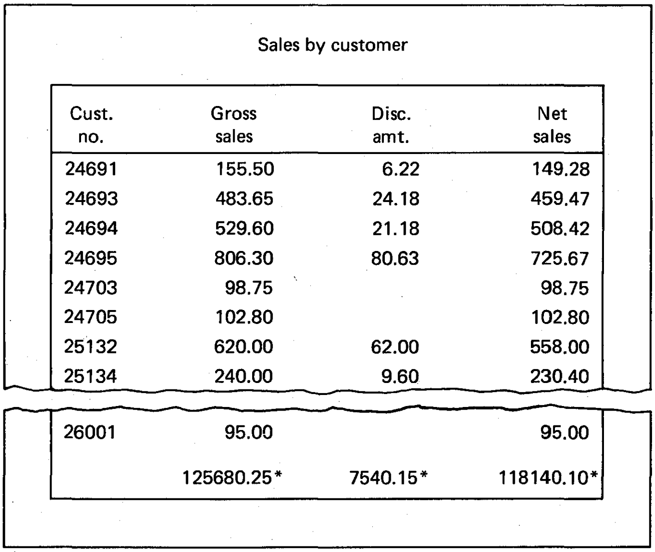


Product sales activity

Prod. no.	Quan.	Unit price	Selling price
1000	125	5.00	625.00
1001	75	4.00	300.00
1002	130	3.00	390.00
1003	50	2.25	112.50
1004	65	2.50	162.50
1005	100	1.20	120.00
1006	118	2.50	295.00
1007	42	5.50	231.00
1008	55	1.65	85.25
1986	430	2.25	967.50
			125680.25*

Figure 8.5 Sorting for printed reports—product number.

formation could include a sales total for some preceding period as an indication of buying trends. For example, sales this year or this month could be compared with a corresponding previous period.



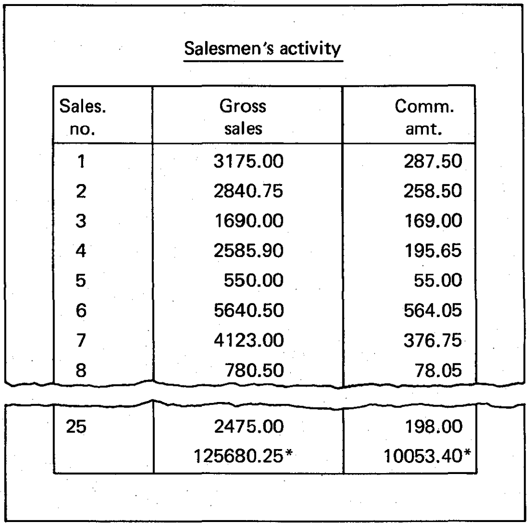
Sales by customer

Cust. no.	Gross sales	Disc. amt.	Net sales
24691	155.50	6.22	149.28
24693	483.65	24.18	459.47
24694	529.60	21.18	508.42
24695	806.30	80.63	725.67
24703	98.75		98.75
24705	102.80		102.80
25132	620.00	62.00	558.00
25134	240.00	9.60	230.40
26001	95.00		95.00
	125680.25*	7540.15*	118140.10*

Figure 8.6 Sorting for printed reports—customer number.

The credit status of a customer's account could also be shown by printing out sales, payments, and balance due.

3. *Salesman Number.* On this report, sales are shown sequenced by salesman number (Figure 8.7). Total sales and resulting commissions are tabulated to show both selling volume and compensation. The report could be the basis of actual commission payments to all salesmen in the company.



Salesmen's activity

Sales. no.	Gross sales	Comm. amt.
1	3175.00	287.50
2	2840.75	258.50
3	1690.00	169.00
4	2585.90	195.65
5	550.00	55.00
6	5640.50	564.05
7	4123.00	376.75
8	780.50	78.05
25	2475.00	198.00
	125680.25*	10053.40*

Figure 8.7 Sorting for printed reports–salesman number.

These examples have been simplified, but in actual practice, the amount of information that can be printed is limited only by the data punched in the sales transaction cards.

Notice that because all of these reports were produced from the same deck of cards, the gross sales amount totals agree. This is an example of proper procedure control using an amount total to check the accuracy of each printed report.

Numerical Sorting

To arrange cards in numerical order, the sorter brush (or sensing head) is set on the right-hand or units column of a numerical field. As cards feed through the machine, they are separated into the various pockets corresponding to numerical punching in that column, as previously explained. The pockets or stackers are arranged in ascending sequence from the operator's right to left as shown schematically in Figure 8.8.

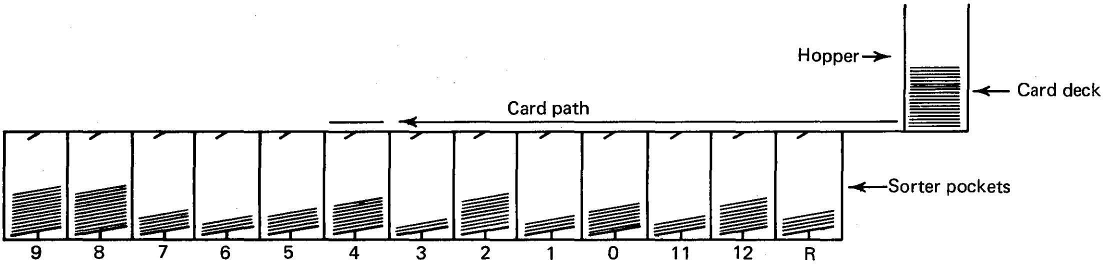


Figure 8.8 80-column sorter pocket arrangement.

The column of numbers in Figure 8.9a represents a small deck of cards that are in random order by a two-digit field. Figure 8.9b

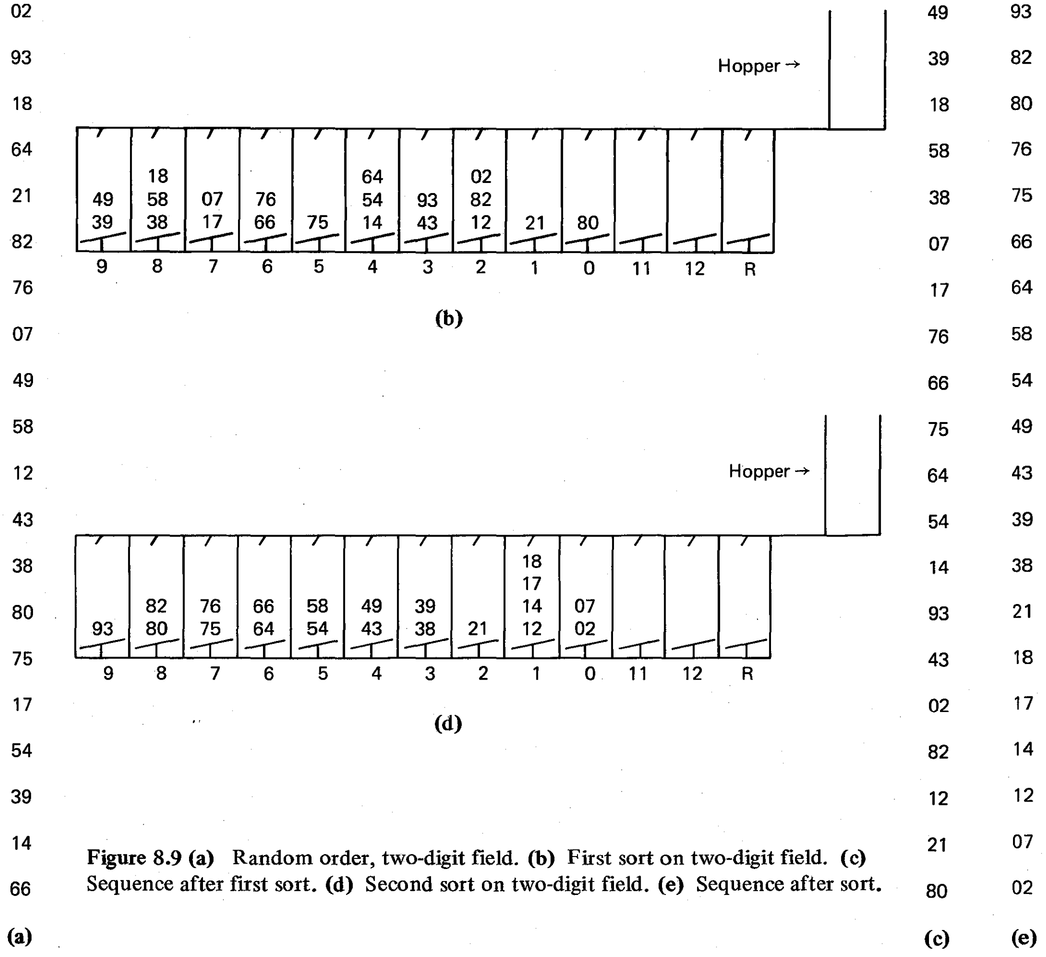


Figure 8.9 (a) Random order, two-digit field. **(b)** First sort on two-digit field. **(c)** Sequence after first sort. **(d)** Second sort on two-digit field. **(e)** Sequence after sort.

shows the arrangement of the same deck of cards in the respective sorter pockets after the first pass through the machine. Cards are directed to the pockets corresponding to the digits punched in the units column.

In actual sorting, the operator now removes the cards, maintaining the sequence obtained on the first sort (Figure 8.9c). Figure 8.9d shows the cards in the pockets after the second pass through the machine. If the cards are removed properly, they will be in sequence as shown by the column of numbers in Figure 8.9e. For larger fields, the same procedure is followed for the third and subsequent columns. After each column is sorted, the cards are in order according to the columns already sorted. Usually, numerical fields have no unpunched columns. Insignificant zeros are punched in columns that precede significant digits. In a strictly numerical sort, no cards should fall into the 11, 12, or reject pockets.

Card Handling

The card sorter, like all other punched-card equipment, is controlled by an operator. And, while the speed of the machine is constant, an average of 25% of total elapsed sorting time normally is required for manual card handling. This includes the movements of removing cards from the pockets or stackers as they are filled, emptying the machine between sorts, checking the accuracy of the sort, loading and unloading file trays, and so on. Good operating procedures will therefore increase efficiency, consequently shortening the elapsed time required to complete any given sort.

By referring to the photograph of the IBM 84 Sorter in Figure 8.1, it can be seen that a flat glass shelf extends over the stackers. This same arrangement is found on all models of IBM sorters. The shelf can be used as a convenient surface on which to ready cards for the file feed or standard card hopper. While a machine is in operation, pockets or stackers can fill very rapidly. Also, cards are hardly ever distributed evenly in the various pockets during a sort. Overflow cards from one or more pockets can be piled on the glass shelf directly above the pocket from which they are removed. When the sort is finished and the machine emptied, the operator knows the exact position in the sequence of each pile of cards on the shelf. Cards are manually stacked facedown. For each subsequent sort, cards are fed into the hopper starting with the card at the *front* or bottom of each stack. This preserves any previously sorted sequence.

Special card trays and dollies can be obtained from a number of manufacturers. The equipment is designed with numbered receptacles or stacking trays to hold cards as they are removed from the machine. Depending upon the volume involved in any one sort—a handful or a number of files—the operator must use some system that will enable him to accurately return cards to the machine in sequence after each sort.

The hopper of the various model sorters is slightly larger inside than the outside dimensions of an 80-column card. Stacks of cards must be *joggled* or aligned carefully so that no protruding edges will jam the deck in the hopper, thereby preventing free access to the feeding knives. Cards are held down in the hopper with a *removable* card weight.

The *throat* of each machine is designed with tolerances that will admit *one* card at a time, but not two. Edges of cards must not be damaged or else feeding will be troublesome, with the possibility of jamming in the transport mechanism or chute blades. Folding cards in the center will thicken the edge exactly at the throat guide with the result that these cards are very difficult to sort accurately.

The well publicized motto "Do not fold, spindle, or mutilate!" is not exactly a joke to the card machine operator who takes every precaution to keep punched-card records in the best condition possible.

An operator quickly acquires the simple skill of card joggling and fanning. To joggle the cards, a convenient size deck is held loosely in one hand while the ends are struck lightly against some solid surface with the other hand. Every card machine except the sorter is equipped with a *joggle plate* for this purpose. The effect is to align all four edges of the deck as evenly as possible.

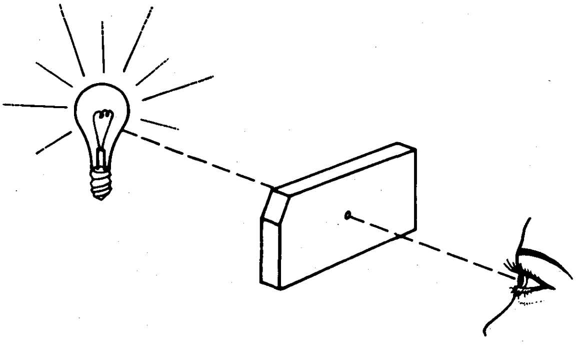


Figure 8.10 Sight checking.

Fanning or riffling the cards is also good operating practice. This is done by holding a small deck by the end in one hand and fanning across the opposite end with the other hand in a brushing motion. Fanning removes static electricity that may form in dry atmospheric conditions and cause cards to stick together. Fanning also removes dust and other foreign particles that sometimes accumulate in a file drawer, particularly when the file is used for reference. Fanning and joggling also show at a glance that all cards face in one direction with corner cuts aligned properly.

Sight Checking

The accuracy of every sort should be checked after removing cards from each pocket. First, the cards from any pocket are joggled into perfect alignment. Then the operator looks through the hole corresponding to the pocket from which the cards were removed. If the cards have been correctly sorted, the holes form a small passage in the deck through which light can be seen (Figure 8.10).

If no light is visible, a missorted card must block the passage because, in that card, a hole is punched in some other position. For example, if the cards come from column 3, all cards in that pocket must have a 3 punched in the column being sorted. If a 2 missorts into the 3 pocket, that card would cut off the *sighting* of light through the three hole in the deck.

A missorted card can be located easily by using a thin metal rod called a *sorting needle* which is small enough to pass through the rectangular holes. The needle is pushed through the passage in the deck (for example, through the 3 hole) until it is blocked by a missorted card. The deck is opened at that point and the card removed. The error card can be manually filed in its proper place if it is not damaged in any way. The operator must also file the missorted card in the current sequence resulting from all previous sorts.

Estimating Sorting Time

Whenever a punched card procedure is designed, consideration must be given to the total elapsed time required to complete an entire procedure cycle. The time required for sorting, usually a significant portion of the estimated time, is always included.

Actual machine time for any given sort can be calculated by multiplying the number of cards in the file times the number of columns to be sorted, then dividing the product by the running speed of the machine (cards per minute). Thus, to sort 20,000 cards on five columns with a 1,000 card-per-minute sorter takes 100 minutes.

$$\frac{\textit{20,000 cards} \times \textit{5 columns}}{\textit{1,000 cards per minute}} = \textit{100 minutes}$$

or 1 hour and 40 minutes

A factor of 25% handling time is usually added to the total machine time. For a novice operator this time may be somewhat longer; for an experienced operator, considerably shorter. Also note that the 25% handling factor is higher for sorting than for most other card processing operations. This is because the stacks of cards from each sorter pocket must be manually removed before the machine can be started on the next pass. Cards must be joggled and fanned manually between each pass and, after each pass, all cards should be sight-checked. Adding the handling factor to the previous example brings the total time to 125 minutes, or two hours and five minutes.

Block Sorting

Block sorting serves two purposes. First, with this technique a large file can be broken down into smaller groups of cards so that several blocks can be sorted at the same time on different machines. Thus, overall elapsed time can be reduced even though the total time required may be slightly more due to increased handling. Second, the next machine operation in the procedure can be started without waiting for an entire file to be sorted.

For example, assume that 20,000 sales transaction cards are to be sorted by customer, a five-digit number. Also assume that the number is punched in columns 21 through 25. To block sort the file, proceed as follows:

1. Sort the entire file on column 21. If there is no sequence to the file that must be preserved, this sort can be done on more than one machine. If there is approximately equal distribution of cards in all pockets, there will be 10 blocks of 2,000 cards each: 2,000 in the 0 pocket, 2,000 in the 1 pocket, 2,000 in the 3 pocket, etc.

2. Take each block of cards and sort in the usual way, from column 25 back to column 22. Two or more blocks can be sorted simultaneously on different machines.

If it takes two hours and five minutes to get the 20,000 cards into customer number sequence using regular sorting techniques, this means that the next machine operation cannot be started until all cards have been sorted as shown in Figure 8.11. If report preparation requires three hours and 20 minutes, the total elapsed time from the start of the sorting operation and the completion of the report would be almost five and one-half hours.

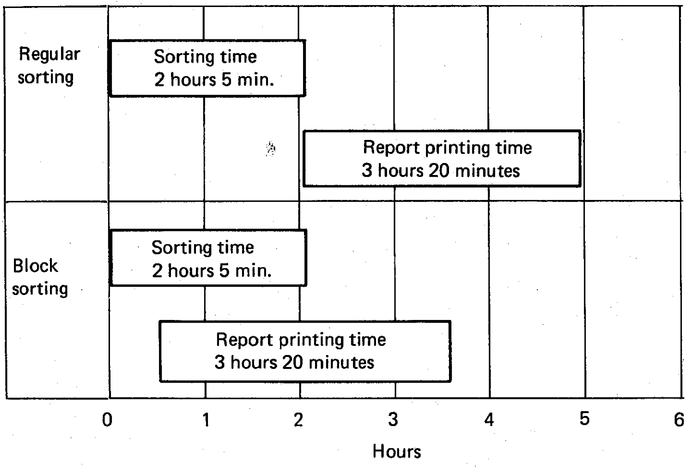


Figure 8.11 Serial sorting vs. block sorting.

Using block sorting methods, the amount of time required to get block 0 in sequence may be calculated as follows:

First sort (column 21) $\quad \dfrac{\textit{20,000 cards} \times \textit{1 column}}{\textit{1,000 cards per minute}} = \textit{20 minutes}$

Second sort, 0 block (columns 25–22) $\quad \dfrac{\textit{2,000 cards} \times \textit{4 columns}}{\textit{1,000 cards per minute}} = \textit{8 minutes}$

Subtotal *28 minutes*

25% handling time *7 minutes*

Total *35 minutes*

Thus, report printing can begin approximately 35 minutes after the beginning of the sort. Printing can continue while the balance of the file is being sorted. With this method, the overall elapsed time is reduced by about one and one-half hours as shown in Figure 8.11.

Alphabetic Sorting

Letters of the alphabet are represented in 80-column cards by both numerical and zone punching in a single column. Therefore, to sort alphabetically, two sorts are required on each column. The first sort is numerical, that is, the column is sorted on numerical punches only. On the second pass, the sort is made on the 0, 11, and 12 punches only.

In normal sorting procedures, all cards are fed to the sorter with the 9 edge toward the throat. By referring to Figures 8.3 and 8.4 it can be seen that when a card passes under the sensing brush (or the diode), the *lowest* hole in the column is read first. A 9 hole is read before an 8, a 6 hole before a 0, and so on. The *first* electrical impluse positions the chute blades to direct the card to the corresponding pocket, regardless of how many *other* holes may be punched in the same column. Any *higher* punching in the column is ignored.

Therefore, when sorting alphabetically, the first pass sorts the cards into numerical sequence by the digit punches. The zone punches are ignored. On the second pass, the sort is made on the 0, 11, and 12 punches in the same column. This is done by disconnecting the digit circuits of the machine. An *alphabetic sorting switch* is provided for this purpose.

After the second sort, the 12 pocket contains the letters A through I, the 11 pocket contains the letters J through R, and the 0 pocket contains the letters S through Z. In alphabetic sorting, as in numerical sorting, the right hand column of the field is sorted first. Successive sorts are made from right to left with the last sort on the leftmost column of the field.

The column of letters in Figure 8.12 represents a small deck of cards in random sequence. The deck is to be sorted alphabetically by a two-column state abbreviation code. Four sorts are required. Each letter in the column is shown with its corresponding punched-hole code. Assume that the field is punched in columns 26 and 27.

Card code	Abbr.		Card code
0-3	T	X	0-7
11-5	N	Y	0-8
11-5	N	J	11-1
0-5	V	T	0-3
12-6	F	L	11-3
0-2	S	C	12-3
11-5	N	H	12-8
0-5	V	A	12-1
0-6	W	Y	11-8
0-2	S	D	12-4
11-5	N	D	12-4
11-5	N	C	12-3
12-7	G	A	12-1
11-3	L	A	12-1
11-4	M	E	12-5
11-4	M	O	11-6
11-9	R	I	12-9
11-7	P	A	12-1

Figure 8.12 Random order, two-character field.

The first sort is made on the numerical portion of column 27. Cards fall into the respective pockets as shown in Figure 8.13a. The second pass is made on the zone portion of column 27. The same sorting procedure is repeated for column 26. The resulting distribution of cards is also shown in Figure 8.13b. The final sequence is represented by the column of abbreviations shown in Figure 8.14.

Column 27 First pass (numerical)

9	8	7	6	5	4	3	2	1	0	11	12
RI	NY NH WY	TX	MO	ME	SD ND	VT FL SC NC		NJ VA GA LA PA			

Second pass (zone)

0	11	12
NY WY TX VT	RI MO FL NJ	NH ME SD ND SC NC VA GA LA PA

(a)

9	8	7	6	5	4	3	2	1	0	11	12
RI		GA PA	WY FL	NY VT NJ NH ND NC VA	MO ME	TX LA	SD SC				

Column 26
First pass
(numerical)

0	11	12
WY VT VA TX SD SC	RI PA NY NJ NH ND NC MO ME LA	GA FL

Second pass
(zone)

(b)

Figure 8.13 First (a) and second (b) sorts on two-character field.

WY
VT
VA
TX
SD
SC
RI
PA
NY
NJ
NH
ND
NC
MO
ME
LA
GA
FL

Figure 8.14 Sequence after second alphabetic sort.

Alphabetic fields usually have blank spaces between words. These unpunched columns cause the cards to fall into the reject pocket on the first sort. The pocket need not be sorted again. Rejected cards are simply placed at the front of the file after the second sort and before proceeding to the next column.

Alphabetic fields can also be block sorted. One method is to sort the leftmost column of the field on the zone portion only. This sort divides the file into three sections by the 0, 11, and 12 punches. Each block can then be sorted separately.

Selection Switches In addition to the alphabetic sorting switch, each type IBM sorter is equipped with twelve single selection switches. There is one switch (or button, depending upon the machine model) for each of the 12 punching rows of an 80-column card. When turned off, a switch suppresses sorting of the corresponding rows. The sorter then rejects all cards punched in that position of a column.

More than one switch can be turned off at a time. It is therefore possible to sort cards punched with more than one digit or zone code into the reject pocket. For example, if the 2 and 3 switches are turned off when sorting numerically on any column, all cards punched with a 2 *or* 3 will sort into the reject pocket. Cards punched with the digits 0, 1, and 4 through 9 will sort normally. The effect is to select all 2s and 3s from the file into one pocket.

Conversely, if all selection switches are turned off *except* any one single switch, the cards with punching in that row will sort to the corresponding pocket while all other cards are rejected. For example, assume that all switches are turned off except the three switch. When cards are sorted on any column, all cards will reject except those punched with the digit 3. Those cards will be *selected* into the 3 pocket. Note that the sequence of all the cards has not been disturbed; they have merely been divided into two separate groups or files.

Selection has many practical uses in punched-card procedures. It provides a means of high-speed access to records with a method of readily classifying information into various predetermined categories. Consider the following three examples.

1. Sales transaction cards may be selected from a file for those customers who have made single purchases of any item of $1,000 or more. If the amount field is in columns 75 through 80 (four digits for dollars and two digits for cents), then selecting 0s in column 75 will remove from the file all cards with an amount less than $1,000. Cards in either or both files can be printed by an accounting machine for further analysis.

2.. Personnel and statistical punched cards are particularly well adapted to selection techniques, especially when coded by various categories. Assume that records are coded 1 for salaried employees, 2 for hourly employees, 3 for commission employees, 4 for salary plus commission, and so on. Separation of employee records by wage class can be done automatically at speeds up to 2,000 cards per minute. Once selected, reports can be printed for analysis.

3. As a result of calculation of amount fields, credit or minus balances often result in accounting procedures. For example, if a customer overpays a bill, the amount due field in his accounts receivable record no longer represents an amount due the creditor, but a refund due the customer. Such minus fields are usually punched with an 11 or X punch over the units digit of the field. By selecting X-punched fields, all refund amounts can be separated from the accounts receivable file for special handling.

Alphabetic Shortcut Method

Using selection switches, the following shortcut method can save approximately 16% of alphabetic sorting time.

For the first sort on each column, the cards are placed in the hopper faceup, 12-edge first, with the 9 selection switch off. Because the cards are entering the sorter 12-edge first, a 12 zone punch is read as a 9, an 11 punch as an 8, and so on. But, because the 9 selection switch is off, the 12 zone punch is not read. Therefore, all letters with a 12 zone punch sort on numerical punches, or what *appear* to be numerical punches to the machine. Cards punched with the letter A (Punched 12-1) fall into the 6 pocket; the Bs (12-2) fall into the 5 pocket; the Cs (12-3) into the 4 pocket, and so on up to the Is (12-9) which fall into the 12 pocket. Cards with 11 and 0 zone punches sort into pockets 8 and 7 respectively.

All cards punched with the letters A through I are completely sorted on this first sort. These cards can now be removed from pockets 6 through 12 and stacked.

The 9 selection switch is now turned on again. The cards remaining in pockets 7 and 8 are sorted separately in the normal manner, facedown, 9-edge toward the machine. At the conclusion of this sort, the previously stacked cards punched with the letters A through I are placed at the front of the deck. The process is repeated for each succeeding column of the sort until the cards are completely arranged in alphabetical sequence.

Special Sorting Devices

Sorters can be equipped with a number of special devices to make the sorting operation more efficient. For example, the IBM 83 Sorter is provided with a five-position *sort selection switch.* The setting of the switch determines the sorting pattern to be followed by the machine, as shown in Figure 8.15. The conventional sorting patterns for either alphabetic or numerical are available with N or Z settings. In addition, three other variations on alphabetic sorts can be used by the operator.

SORT SELECTION SWITCH SETTING	POCKETS												REJECTS REGARDLESS OF EDIT	ERRORS (When Edit or Edit-Stop is ON)
	9	8	7	6	5	4	3	2	1	0	11	12		
Numerical (N)	9	8	7	6	5	4	3	2	1	0	11	12	Blanks	Multiple-punched cards (incl. letters)
Zone (Z)										0	11	12	Any card without a zone punch	Any card with more than one zone punch
Alpha-1 (A-1)	I	H	G	F	E	D	C	B	A	0 S-Z	11 J-R		Blanks and cards with a 12-zone punch but no digit punch. Digits 1 to 9.	Any card with more than one zone punch or with more than one digit punch
Alpha-2 (A-2)	R,Z	Q,Y	P,X	O,W	N,V	M,U	L,T	K,S	J 0-1				Cards with 0 or 11-zone only. Blanks. Letters A to I, and 12-zone spec. char. Digits 1 to 9.	Same as A-1
Alpha-Numerical (A-N)	9	8	7	6	5	4	3	2	1	0 (digit)	11 J-R	12 A-I	Blanks, 0-zone (S-Z)	Same as A-1

This pattern is based on cards being fed face down, 9 edge first.

Figure 8.15 Sorting pattern for IBM 83 Sorter.

Any deviation from a specified pattern can be automatically detected by setting an *edit switch.* For example, when sorting numerically, any multiple-punched cards in the column being sorted are determined to be errors. Also, when sorting by zones, any card with more than one zone punch is treated as an error. Error cards are rejected, that is, they fall into the reject pocket of the machine.

By setting an *edit-stop switch*, whenever error cards reject an error light comes on and card feeding stops. The operator can then remove the error card from the file.

The 83 and 84 Sorters can also be equipped with an *alphabetic sorting* feature. With this feature installed, the machine operates in one of the three possible sorting patterns shown in Figure 8.16.

SORT SELECTION SWITCH SETTING	POCKETS												REJECTS REGARDLESS OF EDIT	ERRORS (When Edit or Edit-Stop is ON)
	9	8	7	6	5	4	3	2	1	0	11	12		
A-1	X	U	R	O	L	I	G	E	C	A	KN QT WZ	BD FH JM PS VY	Cards punched with digits only, zones only, 0-1 combination, or blank	Any card with more than one zone punch or more than one digit punch
A- 2	Z Y X	W V U	T S R	Q P O	N M L	K J I	H G	F E	D C	B A			Same as A-1	Same as A-1
A-N	9	8	7	6	5	4	3	2	1	0	KN QT WZ 11	BD FH JM PS VY 12	Blanks. A,C,E,G,I, L,O,R,U,X and the combination 0-1.	Same as A-1

This pattern is based on cards being fed face down, 9 edge first.

Figure 8.16 Sorting pattern for alphabetic sorting feature.

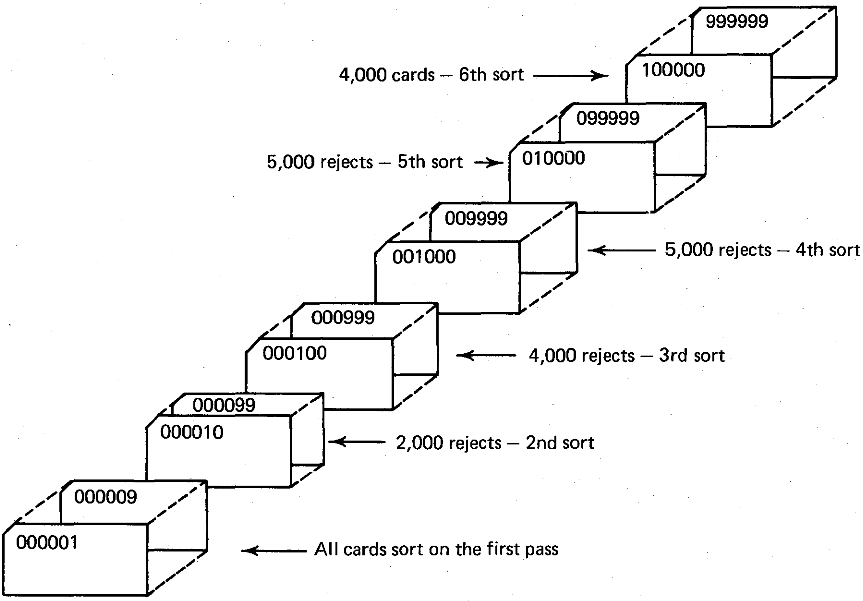
To sort a column alphabetically, the selection switch is first set to A-1. Cards punched A, C, E, G, I, L, O, R, U, and X fall into pockets 0 through 9. The letters B, D, F, H, J, M, P, S, V, and Y fall into the 12 pocket. Letters K, N, Q, T, W, and Z fall into the 11 pocket. Blanks and cards without letter coding fall into the reject pocket.

The selection switch is now set to A-2. It is not necessary to remove cards from pockets 0 through 9. Cards from pocket 12 are now sorted, followed by the cards from pocket 11. Cards fall into their respective pockets and the column is completely sorted.

Occasionally, numerical and alphabetic characters are intermixed in the same field; for example, street and number in address cards. These fields may be sorted in an alphanumerical pattern. Digits 0 through 9 fall into their respective pockets. The letters K, N, Q, T, W, and Z fall into pocket 11. Letters B, D, F, H, J, M, P, S, V, and Y fall into pocket 12. Cards with only an 11 punch fall into the 11 pocket, and 12 zone cards fall into the 12 pocket. All other cards including the letters A, C, E, G, I, L, O, R, U, X, and the 0-1 combinations fall into the reject pocket.

Multiple-Column Selector (IBM 82, 83)

The operation of the sorter can be further enhanced by installing the multiple-column selector feature. Instead of using only one brush to read a single card column, the feature uses 10 brushes that may be set to read any 10 or fewer adjacent card columns. There is also a small control panel on the sorter, 10 column control keys, and control switches. The function and selection to be performed by the device is set up by the operator for the particular procedure involved.



Total number of cards – 20,000 Speed – 1,000 cards/minute

Zero elimination method

Sort	Volume	Time
1	20,000 cards	20 minutes
2	20,000 cards	20 minutes
3	18,000 cards	18 minutes
4	14,000 cards	14 minutes
5	9,000 cards	9 minutes
6	4,000 cards	4 minutes
		85 minutes
	25% handling time	20 minutes
	Total	105 minutes

Regular method

$$\frac{\text{20,000 cards} \times \text{6 columns}}{\text{1,000 cards per minute}} = \text{120 minutes}$$

25% handling time	30 minutes
Total time	150 minutes

Figure 8.17 Zero elimination method of numerical sorting.

Multiple-column selection selects into a specified pocket those cards punched with a predetermined numerical or alphabetic code in 10 or fewer consecutive columns. For example, all cards with city names like Rochester, Syracuse, or Albany can be selected from a given file. The sequence of the remaining cards is not disturbed.

In a marketing situation for a business, the feature could be useful in identifying customers in given locations from a punched-card name and address file. For example, all customers in Rochester could be listed by an accounting machine in planning a sales campaign in that city.

Other variations of multiple-column selection are possible with this feature. *Common-digit selection* sorts out all cards with one or more common digit punches in ten or fewer consecutive columns.

Zero elimination shortens the time required for sorting numerically by rejecting those cards that require no further sorting on insignificant zero to the left of the field. For example, while sorting on a six-digit part number field, number 000065 would reject on the third sort. A card punched 000125 would reject on the fourth sort. Rejected cards are placed in the front of the file, first rejects first, second rejects second, and so on. A schematic of this method is shown in Figure 8.17. In the illustration, it is assumed that the number of cards given will reject during each of the six sorts. The estimated time for each sort is also shown and the total time is compared to the time required to sort the file without the zero elimination feature. A 25% allowance for handling time is added for both methods. Use of the feature saves approximately 45 minutes elapsed time on one machine.

A *length of field* selection can also be made. Cards are distributed by pocket as determined by the *last* significant column punched in the field, regardless of the remaining spaces to the left. When sorting alphabetically, short names like Rye and Olean that require fewer sorting passes can be separated from longer names like Kalamazoo and Albuquerque. Selection can be done on one pass through the machine.

IBM 5486 Card Sorter

The IBM 5486 Card Sorter is designed to sort 96-column cards. As a compact, tabletop machine, the 5486 can be operated from a seated position. The card sorter is an auxiliary unit of the IBM System/3 small computer.

The 5486 operates in much the same manner as the 80-column machines described in the previous section. However, the sorting patterns and procedures are considerably different. The basic features of the machine are shown in Figure 8.18.

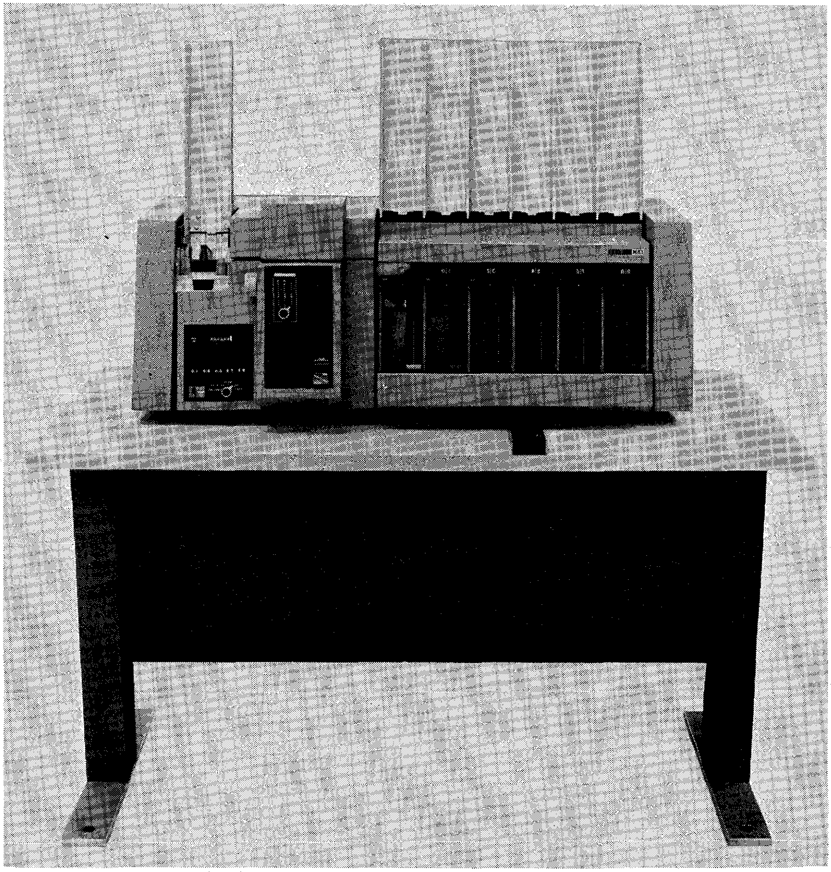


Figure 8.18 IBM 5486 card sorter.

The card hopper is located at the upper right corner and has a capacity of approximately 2,000 cards. The cards are placed face-down in the hopper, top edge toward the machine. Cards are fed from the hopper, one at a time, from the bottom (first card) of the deck. Two model machines are available from IBM that operate at speeds of 1,000 and 1,500 cards per minute.

The 96-column cards move from the hopper to a read station where a beam of light shines through the holes punched in the card and strikes an electronic device. The pattern of holes is sensed to direct the card to one of six pockets. Pockets are labeled 9/8, 7/6, 5/4, 3/2, 1/0, and R (reject). Each pocket holds about 600 cards. The sorter stops automatically when a pocket is full.

A sorting tray is provided at the top of the machine. When a pocket is full, cards are removed by the operator and stacked in the trays. There is one tray for each of the pockets.

Numerical Sorting: 96-Column Cards

Numerical sorting begins with the right-hand, or units digit of the field. With 96-column cards, two selections must be made to locate the proper column: the tier and the column. A card column selector knob is furnished for this purpose on an operator's panel.

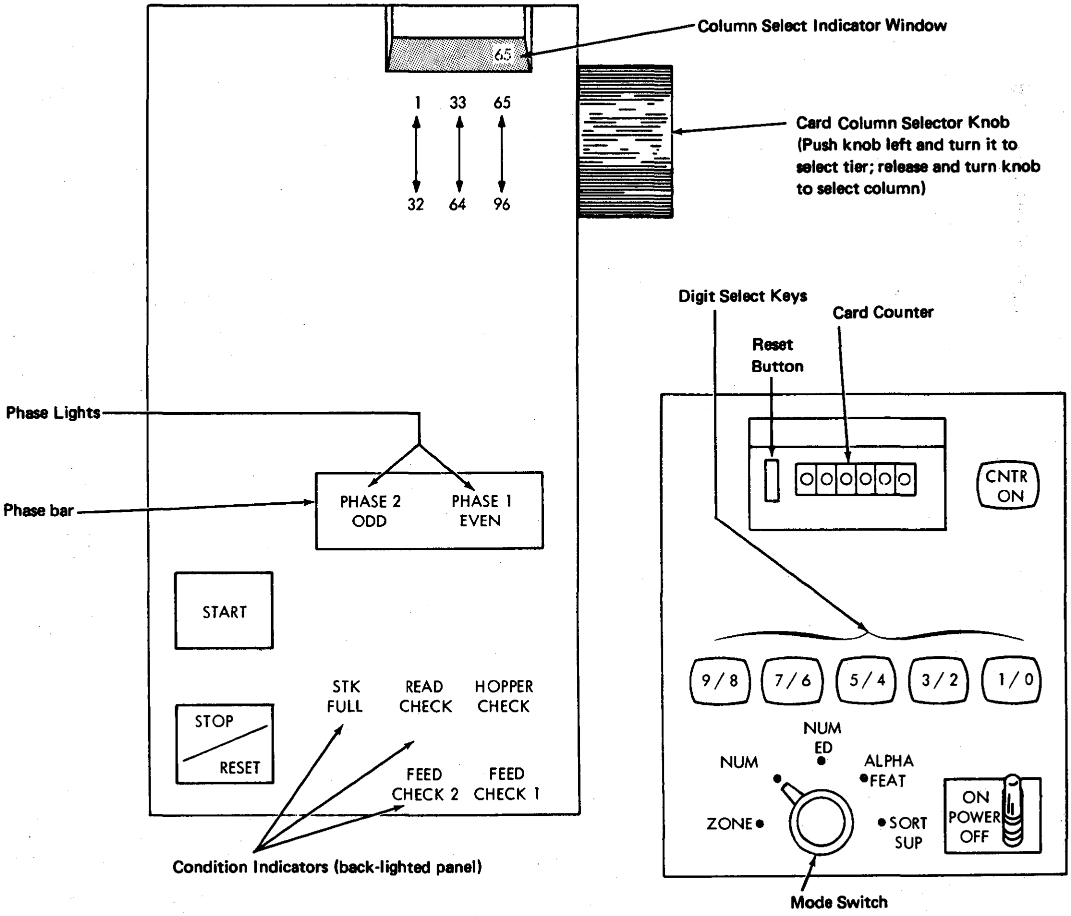


Figure 8.19 5486 operator's panel.

A schematic of the panel is shown in Figure 8.19. The number of the column selected shows in a small indicator window.

Because there are only five numerical pockets in the machine, numerical sorting is done in two phases. In phase 1, all even-numbered digits sort and all odd-numbered digits reject. Thus, 0s fall into the 1/0 pocket, 1s into the reject pocket, 2s into the 3/2 pocket, the 3s reject, and so on. Sorted cards are left in the pockets at the end of phase 1.

A lighted phase bar is provided to change the machine to phase 2 for the second part of the numerical sort. In phase 2, the cards from the reject pocket are placed in the hopper. These cards now contain only odd-numbered digits which sort into their respective pockets on top of the even-numbered digits from phase 1. Digit 1s sort into the 1/0 pocket, 3s into the 3/2 pocket, and so on. The completed pattern is shown in Figure 8.20.

Phases of Sort	Pockets					
	9/8	7/6	5/4	3/2	1/0	Reject
After Phase 2 Sort	9	7	5	3	1	Blanks and Special Characters
After Phase 1 Sort	8	6	4	2	0	All Odd-Numbered Digits, Blanks, and Special Characters

Figure 8.20 5486 numerical sorting pattern.

At the conclusion of Phase 2, cards are in numerical sequence by the selected column. The procedure is repeated, two sorts on each column, until the leftmost column of the field has been sorted. As a sort proceeds, the operator must be sure that cards are taken from the pockets or tray in sequence. The bottom card of the stack is always first into the hopper.

During numerical sorting, the *mode switch* on the operator's panel is turned to *numerical.*

Alphabetic Sorting: 96-Column Cards

To sort 96-column cards alphabetically, the numerical portion of the punched hole code is sorted first, as just described. After phase 2 of the numerical sort, the mode switch is set to *zone.* A third pass is made on the same column.

Cards from the 1/0 pocket go into the hopper first, cards from the 3/2 pocket next, then the 5/4 pocket, and so on. Cards left in the reject pocket after phase 2 are not sorted on this pass because the column is blank. After the zone sort, cards are distributed in the following order: the letters A through I are in pocket 3/2, J through R are in pocket 5/4, S through Z in pocket 7/6, and all digits are in pocket 9/8. Rejected cards are placed in front of the deck before proceeding to the next left column of the field. Notice that this method sorts the field correctly even though there may be interspersed alphabetic and numerical data. This alphameric sort pattern is shown in Figure 8.21.

Phases of Sort	Pockets					
	9/8	7/6	5/4	3/2	1/0	Reject
After Zone Sort	0-9	S-Z	}, J-R	A-I		Blanks and Special Characters
After Phase 2 Sort	I,R,Z 9	G,P,X 7	E,N,V 5	C,L,T 3	A,J 1	Blanks and Special Characters
After Phase 1 Sort	H,Q,Y 8	F,O,W 6	D,M,U 4	B,K,S 2	} 0	1,3,5,7,9,A,J,C,L,T, E,N,V,G,P,X,I,R,Z, Blanks and Special Characters

Note: Right Brace (}) is sorted as indicated ,not as a special character.

Figure 8.21 5486 alphabetic sorting pattern.

When only alphabetic information is to be sorted, the 5486 Card Sorter can be equipped with an *alphabetic sort* feature. With this feature, cards can be arranged in alphabetic sequence in two passes through the machine on each column, instead of three. The

Phases of Sort	Pockets					
	9/8	7/6	5/4	3/2	1/0	Reject
After Phase 2 Sort		W	Q	K		Numerics, Special Characters, and Blanks
		V	P	J	E	
		U	O	I	D	
	Z	T	N	H	C	
	Y	S	M	G	B	
	X	R	L	F	A	
After Phase 1 Sort				Z	Y	X
	W	V	U	T	S	R
	Q	P	O	N	M	L
	K	J	I	H	G	F
	E	D	C	B	A	Numerics, Special Characters, and Blanks

Figure 8.22 5486 alphabetic sorting pattern–special feature.

feature is activated by setting the mode switch on the operator's panel to the *alpha* position.

Sorting begins on the rightmost column of the field. The tier and column to be sorted are selected by the operator with the column selector knob. The phase bar is pressed to begin the phase 1 sort. The resulting pattern of pocket distribution is shown in Figure 8.22. After phase 1, the cards are removed from the pockets and replaced in the hopper: pocket R first, pocket 1/0 on top of pocket R, and so on. After phase 2 is completed, cards are in alphabetic sequence by the selected column. Note that digits, special characters, and blanks are in the reject pocket. Numerical punching is not intended to be sorted with this feature.

The following example shows the actual procedure for sorting on a two-position alphabetic field. The column of letters in Figure 8.23 is intended to represent a small deck of 96-column cards to be sorted alphabetically by state abbreviation code. The corresponding card code is shown beside each letter. The abbreviations

Card code						State abbr.		Card code					
	A –			2	1	T	X		A –		4	2	1
B	–		4		1	N	Y		A –	8			
B	–		4		1	N	J	B	–				1
	A –		4		1	V	T		A –			2	1
B	A –		4	2		F	L	B	–			2	1
	A –			2		S	C	B	A –			2	1
B	–		4		1	N	H	B	A –	8			
	A –		4		1	V	A	B	A –				1
	A –		4	2		W	Y		A –	8			
	A –			2		S	D	B	A –		4		
B	–		4		1	N	D	B	A –		4		
B	–		4		1	N	C	B	A –			2	1
B	A –		4	2	1	G	A	B	A –				1
B	–			2	1	L	A	B	A –				1
B	–		4			M	E	B	A –		4		1
B	–		4			M	O	B	–		4	2	
B	–	8			1	R	I	B	A –	8			1
B	–		4	2	1	P	A	B	A –				1

Figure 8.23 Random order. Two-character field with 96-column card codes.

are repeated from the previous example in Figure 8.12. Assume the field is punched in columns 26 and 27.

Pockets

Column 27	9/8	7/6	5/4	3/2	1/0	Rej.
Phase 1	ME	NJ SD ND	SC NC RI MO	VT NH	NY VA WY GA LA PA	TX FL
Phase 2	NY WY TX	VT	MO FL	NJ RI NH	ME SD ND SC NC VA GA LA PA	

Column 26	9/8	7/6	5/4	3/2	1/0	Rej.
Phase 1	WY	VT VA PA		NY TX NJ NH ND NC	MO ME SD SC GA	FL RI LA
Phase 2		WY VT VA TX SD SC RI	PA NY NJ NH ND NC MO ME LA	GA FL		

Figure 8.24 Alphabetic sorting. 96-column card and 5486 special feature.

First, the column selector knob is set to tier 1, column 27. The column is sorted on phase 1. The resulting distribution of cards is shown in Figure 8.24a. All cards are removed from the pockets and replaced in the hopper in sequence: rejects first, 1/0 pocket next, etc.

Figure 8.24b shows the distribution of cards after the phase 2 sort on column 27. The entire procedure is repeated on column 26. Figure 8.24c shows the distribution of cards after phase 1; Figure 8.24d shows the cards after phase 2. Figure 8.25 shows the final sequence after the sort has been completed.

Counter On (Special Feature)

The 5486 Card Sorter can be equipped with an auxiliary card counter to count the number of cards being sorted. The counter is located on the operator's panel.

Digit Select (Special Feature)

Five keys on the operator's panel can be used to select cards punched with specified digits. For example, pressing the 5/4 key will select all cards punched with a 4 into the 5/4 pocket during phase 1 sort. All other cards are sorted into the reject pocket.

During phase 2 sort, all cards punched with a 5 will sort into the 5/4 pocket. All other cards will be rejected. One or all keys can be pressed during a sort operation.

Column and tier for the specified digit are selected with the card column selector knob.

WY
VT
VA
TX
SD
SC
RI
PA
NY
NJ
NH
ND
NL
MO
ME
LA
GA
FL

Figure 8.25 Alphabetic sorting. Final sequence of two-character field.

Questions and Exercises

1. Explain why card decks are placed in the sorting machine "facedown, 9-edge toward the machine."
2. Why do sorters normally read only one card column at a time as cards pass through the reading and selecting mechanism?
3. Outline your concept of good sorter operating practices.
4. Estimate 80-column sorting times for the following files:

Model 82 Sorter:	50,000 cards	6-digit field
Model 83 Sorter:	75,000 cards	4-character field
Model 84 Sorter:	106,000 cards	8-digit field
Model 84 Sorter:	95,000 cards	3-character field

5. If each of the above are block-sorted, show estimated elapsed time for each first block to be in sequence.
6. What characteristics of the files must be known before you can make an accurate estimate in answer to question 5?
7. How does the multiple-column selector shorten the sorting time?
8. If equipment is available, sort your requisition cards prepared in chapter 6 in preparation for a report showing item, description, unit price, total cost, and department using. If equipment is not available, cards may be manually sorted to simulate machine action.

9 FILE MAINTENANCE

It has been shown that many types of punched-card records can be classified as *historical*. That is, the card is a coded representation of a transaction that has been completed some time in the past. Examples of this type of record have been mentioned as sales transactions, inventory activity records, payroll earnings, census data, and so on.

In some cases, a record may be created in *anticipation* of a transaction. For billing operations, cards are often prepunched in advance of actual payment. When payment is received, the cards then become historical data to be used in updating the accounts receivable file.

Utility bills are a common example of such records. The bill is submitted to the consumer of power, gas, or telephone service in the form of a prepunched card. The record contains all the pertinent data for the individual account: customer identification, type and amount of service, and the charge. The customer returns the card with payment at which time the accounts receivable is updated to reflect the transaction, now historical data.

However, this chapter will discuss the handling and processing of information that might be classified as *status records.* These are records which represent the present status or state of some situation. And in every case, the data processing procedure is concerned mainly with keeping records up to date with changes in the situation that the file represents. This operation is called *file maintenance.*

For example, name and address files must be continually updated to reflect the corresponding changes of location by customers, subscribers, members, etc. Insurance files must keep up with the changing status of policyholders with respect to births, deaths, marriages, property loss and damage, accident liability, amount of insurance, and premiums. Files of student records must be continually updated to show academic standing, subjects completed, credits earned, etc.

Inventory files change as items are taken from the stock room, ordered, and put back. Accounts receivable files change as customers pay their bills and charge new purchases. Payroll files change as earnings are calculated and accumulated to new year-to-date earnings and tax balances.

Two or more files must be handled in an operation of file maintenance. One file, the *master* file, is invariably set up in the approved collating sequence of the system. In punched-card accounting, this is preferably numerically ascending. The sequence is established on an identifying or *control* field of the records. The field usually contains such identification as customer number, policy number, item number, or employee number.

A master file contains additional data related to the control field; for example, value or quantity, balance due, number of items in stock, policy loan value, hours worked, year-to-date earnings, and so on.

Transactions that affect the status of the master file make up the second file in the operation. Records in this *detail* file must be identified in the same manner as the master file and must be sequenced in the same order by the same control field.

There are three basic methods of performing the operations of file maintenance:

1. *Substitution* of a completely new and updated master record. This means actually taking the old record out of the file and replacing it with a new one.

2. *Transcription* of new or additional information to the master record. That is, the old record is updated with the addition of *new* data. For punched cards, this means punching an additional field and assumes that space is available in the card before the data is added. This method is better adapted to computer file-handling practices.

3. *Summarization* of merged master and detail files. Information is calculated or accumulated from both files to update the master card. This method is most commonly used to adjust amount or quantity balances in the master record. For example:

Previous balance (master) ± *transactions (detail)* =

new balance (updated master record)

With 80-column cards, summarizing is done on an accounting machine. The updated master is produced by a special *summary punch* connected to the machine by cable. Reports or other documents are printed as by-products of the process and also as a means of exercising procedure control over the file maintenance operation.

With the System/3, reading and punching of 96-column cards is done by a multifunction card unit. Reports are prepared with a printer that operates under control of the central computer.

IBM 88 Collator

As a review of the various ways of manipulating card files by machine, the following sections will describe the operation of the IBM 88 Collator (Figure 9.1). This extremely versatile machine automatically performs almost every operation of file handling, including record selection, sequence checking, merging, matching, and editing.

Because the machine is designed to read and compare two files of 80-column cards simultaneously, it is equipped with two feeds, a primary and secondary. Normally, the primary feed is used to read a master file while the secondary feed reads the detail file. Each feed operates at a speed of 650 cards per minute. When both are in use, a maximum of 78,000 cards an hour can be put through the machine. The volume varies, however, depending upon the operation being performed. Both feeds can be equipped with a file-feeding device, each with a capacity of approximately 3,600 cards. Figure 9.2 shows the file feed in place on the primary card hopper.

Figure 9.3 is a schematic diagram of card paths through the collator. Five pockets are provided to stack cards as merged, matched, or selected. Because decks of cards are transported from opposite directions into one file, they must be placed in the hoppers differently. In the primary hopper, cards are placed face-down, 9-edge toward the machine. In the secondary hopper, they

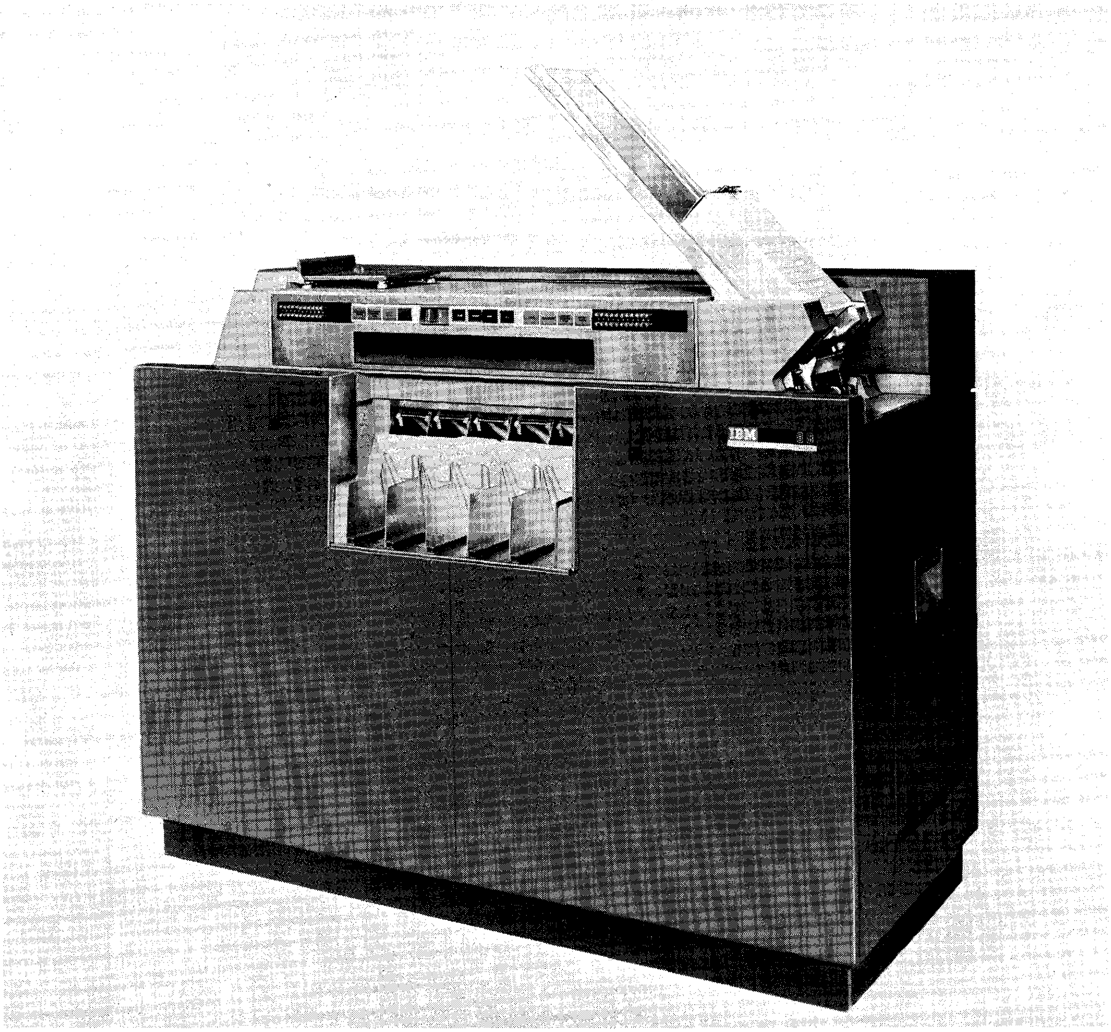


Figure 9.1 IBM 88 Collator.

are placed facedown, 12-edge toward the machine. Figure 9.4 shows the card paths as they would appear looking down from the top of the machine.

As cards are transported one at a time from the primary feed hopper, they pass two 80-column reading stations: primary sequence read and then primary read. Each reading station is equipped with 80 reading brushes. Cards pass between the brushes and a metal roller, row by row, so that any column in the card can be read.

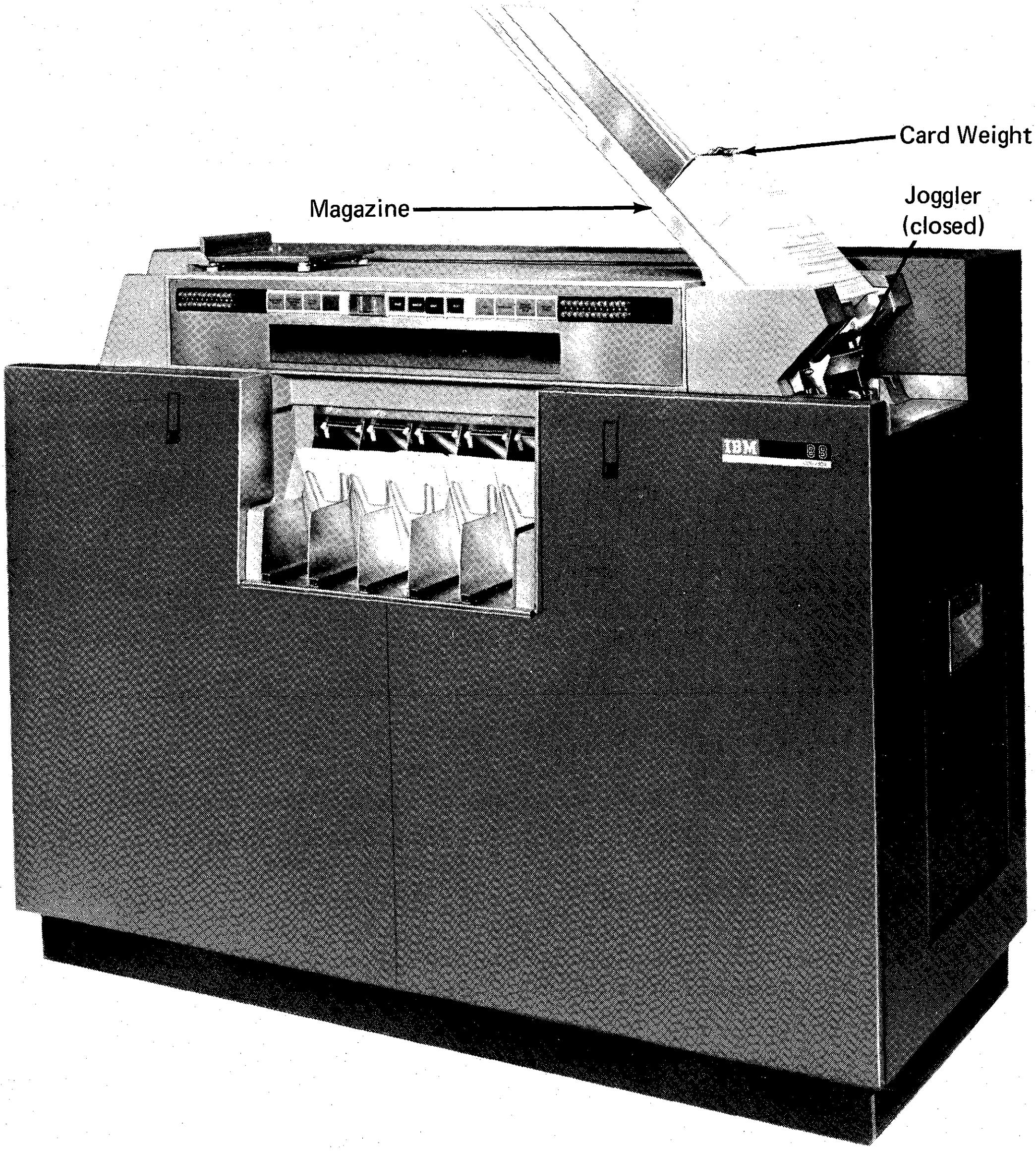


Figure 9.2 File feed device on primary hopper.

Unlike the sorter, which has only one reading brush, the resulting multiple electrical impulses can be sensed by the machine as a complete number or name, not just as a single digit or zone punch. In all operations of card matching and merging, the machine is set up beforehand to read a particular sequenced control

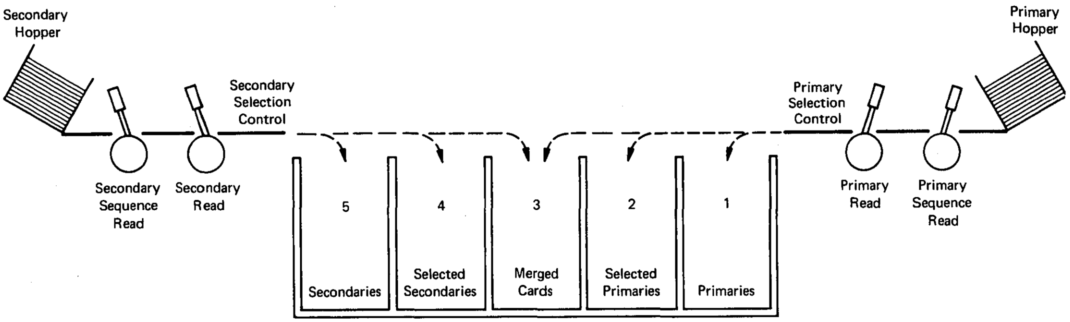


Figure 9.3 Card paths through the collator.

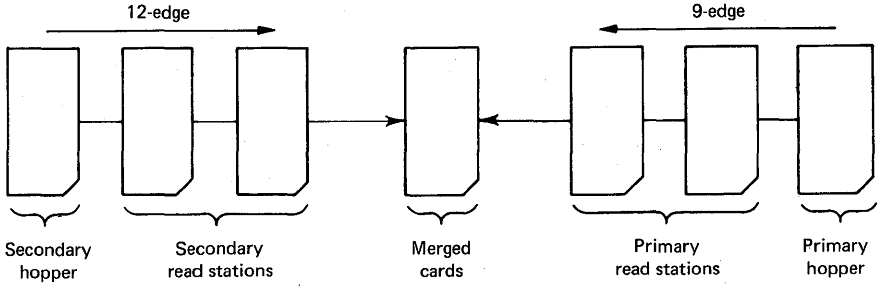


Figure 9.4 Card feeding through the collator.

field of the file. The field to be read and the type of operation to be performed are prewired into a *control panel.* The panel is inserted by the operator. Detailed instructions for wiring the control panel are given in the manufacturer's manuals furnished by IBM. Some further explanation of control panel wiring is also presented later in this text.

Sequence Checking

By proper control panel wiring, the control field of the cards passing through the primary feed can be read into a *primary sequence* unit. Cards pass the two reading stations in exactly-timed feeding cycles so that the control fields may be compared in the sequence unit (Figure 9.5). That is, the selected card field sensed at the primary read station may be compared with the same field being read at the primary sequence station. Because a file should normally be in ascending sequence, the cards are in order if any card is either higher than or equal to the card ahead. However, if a card is lower than a card ahead, an error in sequence is indicated. The error is recognized as a *step-down* or low-sequence condition.

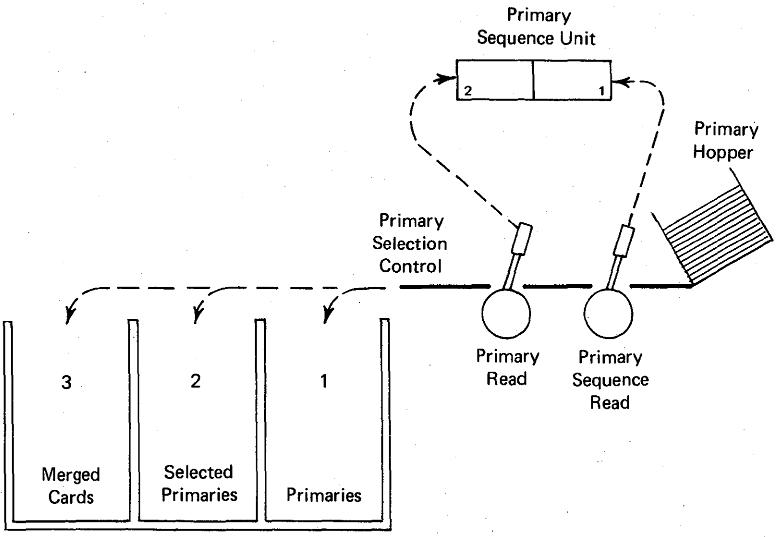


Figure 9.5 Primary sequence unit.

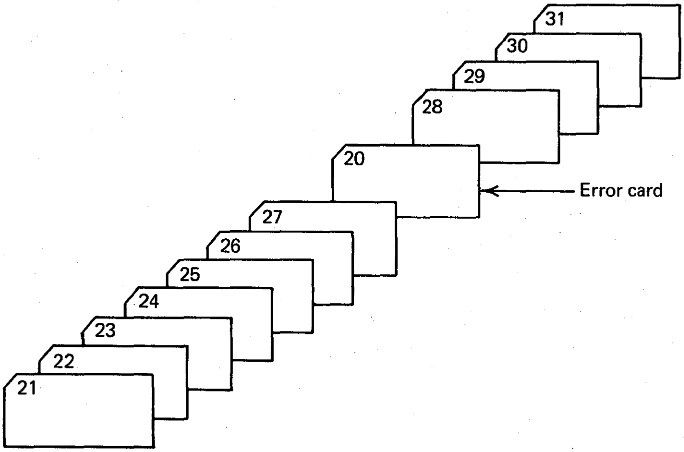


Figure 9.6 Sequence error. Step-down error card.

Figure 9.6 shows a file of cards in ascending numerical sequence with one card (20) out of order. As the cards are fed through the primary feed of the collator, a step-down occurs before the error card as shown in Figure 9.7. This error condition can be wired on the control panel to stop the machine. The cards are removed by the operator and manually arranged to restore the proper sequence. In this case, the error card should be placed at the front of the deck.

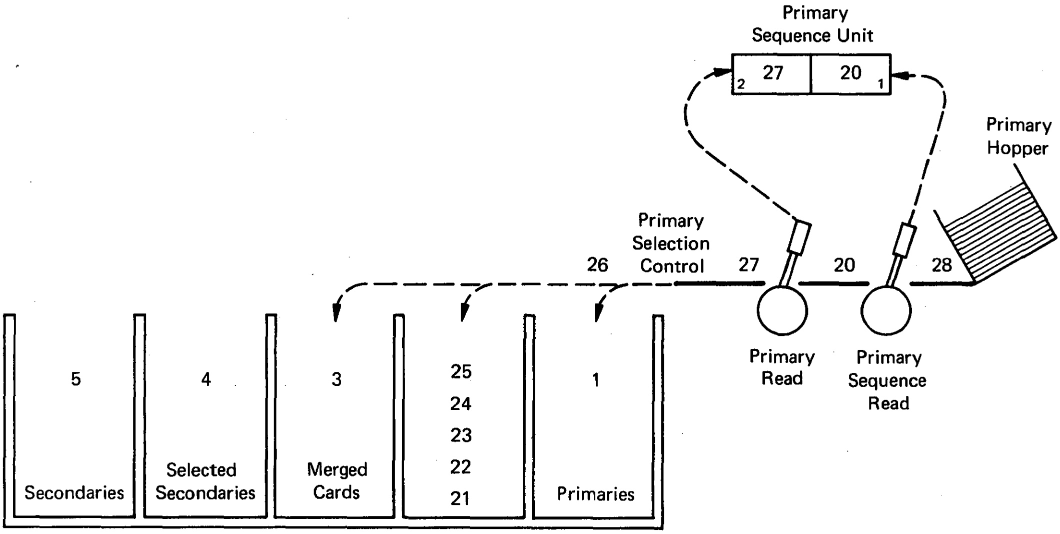


Figure 9.7 Step-down. Primary sequence unit.

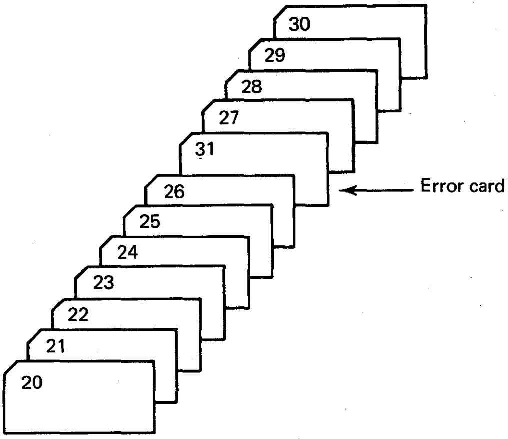


Figure 9.8 Ascending sequence error before step-down.

Figure 9.8 shows an error condition where the step-down in sequence occurs after the error card (31). Figure 9.9 shows a sequence where the card *after* the step-down is in error (23). In addition, the two cards following the first error card are also out of sequence (24 and 25). Notice that in this situation, the machine can indicate only one error, the first step-down. Succeeding cards are in ascending sequence even though they may be lower in order than the cards which have now passed both reading stations.

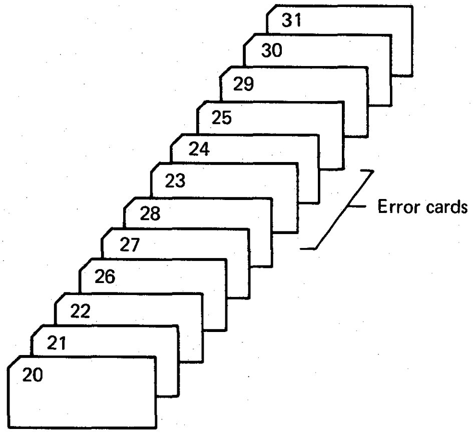


Figure 9.9 Ascending sequence error–three cards.

The collator can also be set up to insert a blank error indicator card into the file whenever a step-down condition occurs. Checking is done in one feed while any required blanks are inserted from the opposite feed. Machine operation is not interrupted and correction of errors must be done manually after the file has been processed. Indicator cards should be of contrasting color and with an opposite corner cut from the original file cards.

Because the sequence checking just illustrated is electromechanical, it is possible to check a file only in the order that is valid to the machine. The sequence must follow the order that is designed and built into the mechanism by the manufacturer.

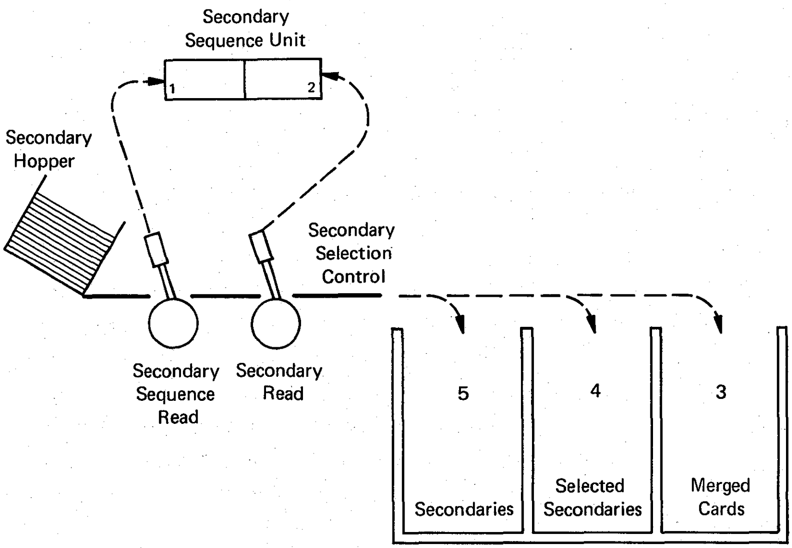


Figure 9.10 Secondary sequence unit.

The 88 Collator can be used to check the sequence of a file in either primary or secondary feed. The machine is equipped with a secondary sequence unit as shown in Figure 9.10. When equipped with an alphabetic collating feature, the machine can process alphabetic information–selecting, checking, merging, and matching cards by names or titles. Each letter is treated as a two-digit number; that is, the numerical portion of a letter is read as punched while the zone portion is read as a 1, 2, or 3. Sequence checking, either alphabetical or numerical, can be done in combination with other operations of file processing.

Sequence checking of descending orders is also possible by proper wiring of the control panel.

Merging Two Files

Each set of 80 reading brushes, primary and secondary, can also be connected to a *comparing unit* in the collator with control panel wiring. Figure 9.11 is a schematic of this arrangement. As cards pass the two reading stations in sequence, the control field from the primary read is compared with the control field from the secondary reading station. One of three possible conditions can result from the comparison:

1. The primary card is *lower* than the secondary.

2. The primary card is *equal* to the secondary. The two cards match.

3. The secondary card is *lower* than the primary.

The low primary, low secondary, or equal comparison is available as a timed electrical impulse to actuate the card feeding and selecting mechanisms. *How* the comparisons actuate machine

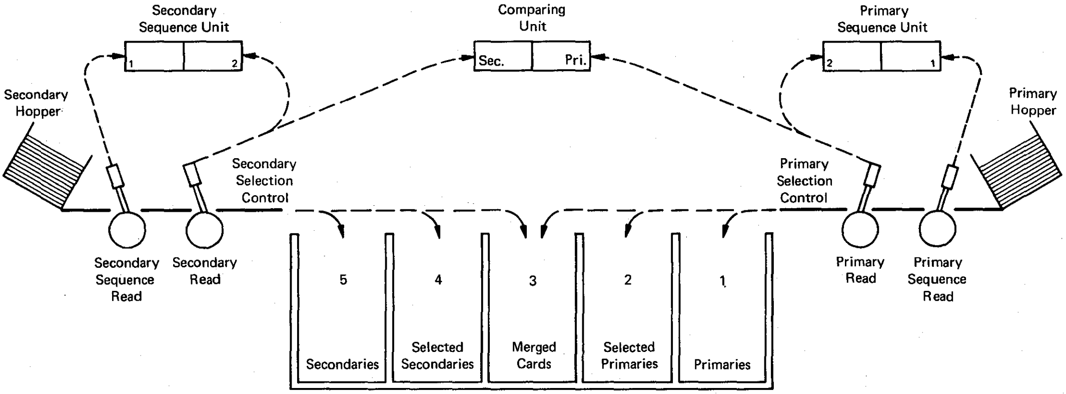


Figure 9.11 Comparing unit.

operation is a function of control panel wiring. Actually, both of the comparing units and the sequence units are used to compare four cards simultaneously, one at each of the reading and sequence stations. The following example has been somewhat simplified to show only the results of comparisons.

Figure 9.12 is a schematic showing a master file and a detail file to be merged in ascending numerical sequence. The master file is placed in the primary feed; details are in the secondary.

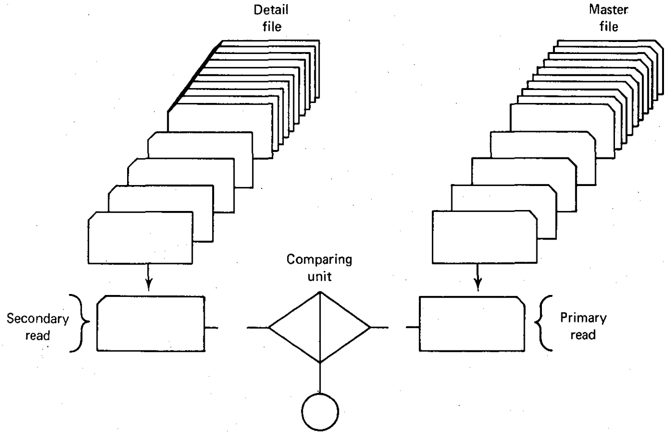


Figure 9.12 Schematic. File merging.

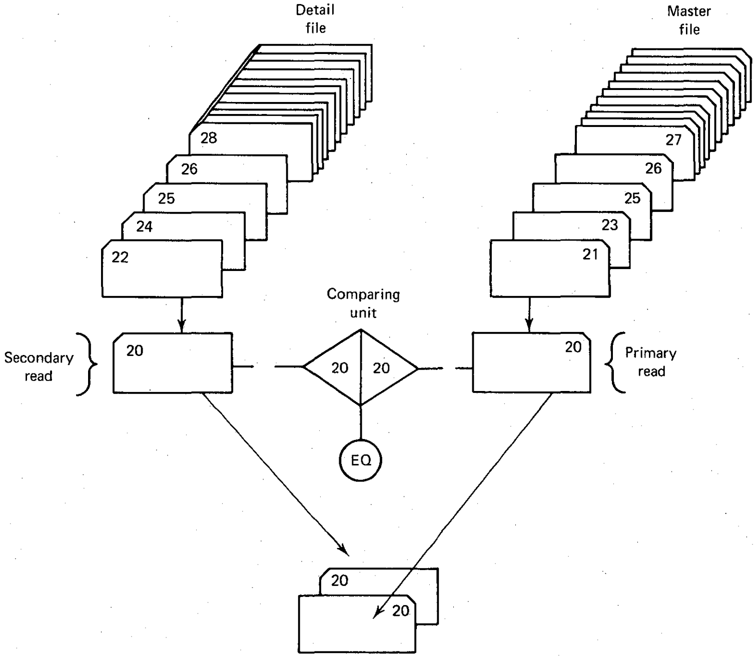


Figure 9.13 File merging. Equal comparison.

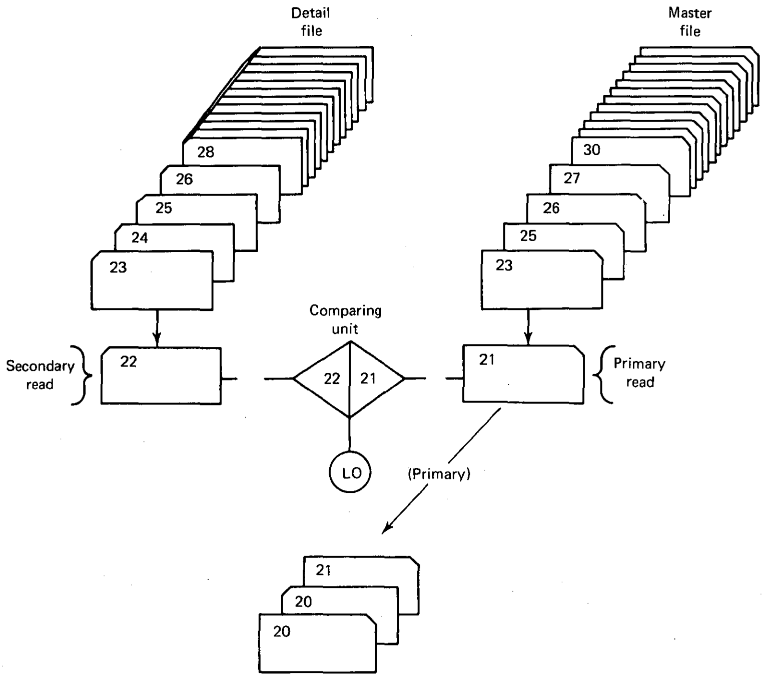


Figure 9.14 File merging. Low primary.

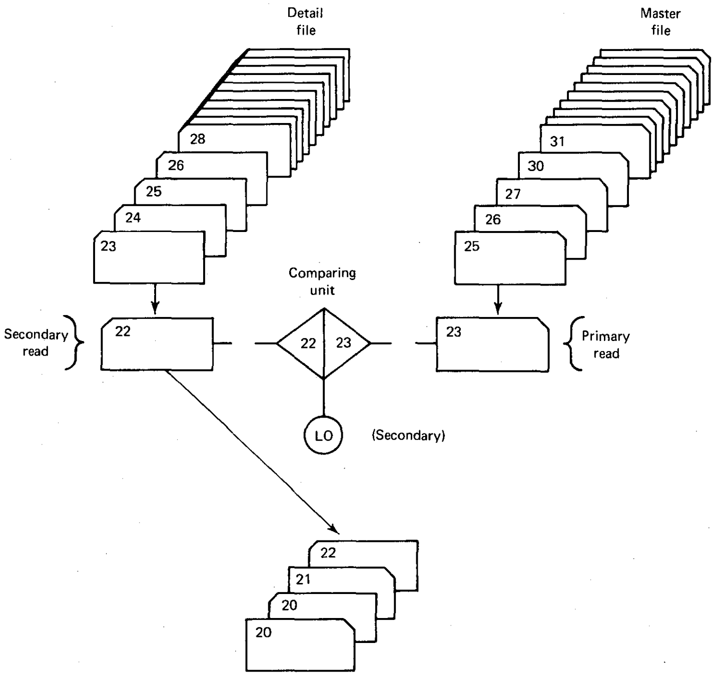


Figure 9.15 File merging. Low secondary.

The first primary card (20) is compared against the first secondary card (20). The comparison is equal as shown in Figure 9.13, and the two cards are fed into the selected pocket. The primary card is first into the pocket with the matching secondary behind it. Succeeding cards in both files are read on the next feeding cycle.

The next comparison shows that the following primary card (21) is lower than the next secondary (22) as shown in Figure 9.14. Therefore, the primary card is fed into the pocket while the secondary is held. On the next cycle, the following secondary (22) is lower than the primary (23) as shown in Figure 9.15. A low secondary comparison results and the secondary card is fed into the pocket.

Machine operation continues automatically until all the cards are merged in sequence.

Merging with Selection

Whenever master and detail files are merged together in preparation for some further processing, it is usually necessary to be sure all details are filed behind a corresponding master card. The details must *match* the following by the sequenced control field of the master file: employee number, account number, policy number, part number, etc.

For example, in the preparation of a payroll, it is always necessary to identify each employee by name. Because such identifying data is constant and does not change from one pay period to another, a master name file is usually kept as a permanent record of the identity of each employee. Other information can include address, number of dependents, sex, age, and so on. Also, the limited capacity of cards usually does not leave room for name and address in the same card with earnings, hourly rate, hours worked, and other earnings information for a particular pay period.

The master file is kept most conveniently in sequence by employee number. To prepare for merging the current earnings records with the master name file, it is first necessary to sort all current pay cards by employee number. The detail transactions may then be merged with the master records to produce paychecks, earnings statements, pay registers, and other reports of year-to-date earnings and tax amounts. The two files may be merged automatically by a collator as previously described.

No employee's current earnings record should reach the final payroll processing run without a name card. Therefore, during the

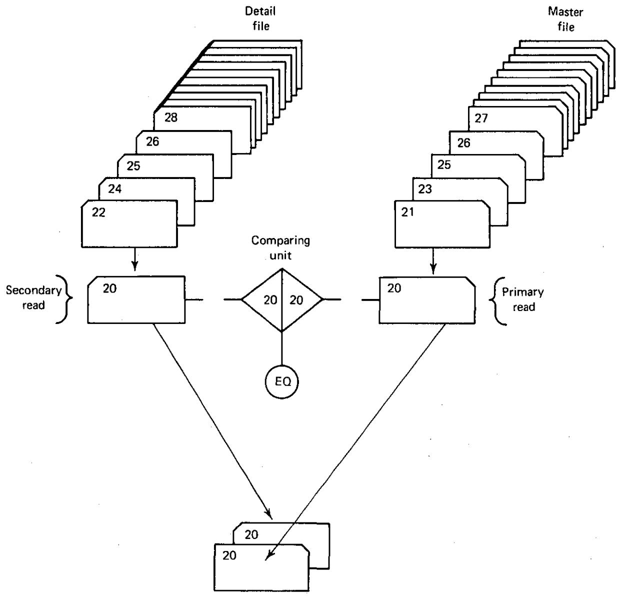


Figure 9.16 File merging with selection. Equal comparison.

merging operation, any current pay cards that do not match a master name card should be selected by the machine for investigation. Cards so selected can mean a new employee on the payroll with incomplete records or a lost or misplaced name card. The selection of unmatched cards is also an important element of procedure control. It can be used as a check that no unauthorized employees are paid.

For the collator run, the master file is placed in the primary feed; current pay cards go in the secondary. All equal cards are merged into a single pocket. Unmatched secondary cards are selected into a separate pocket. Figure 9.16 shows the merging operation. The first cards (20) from the two files match, producing an equal comparison. These cards are fed into the merge pocket.

On the next comparison (Figure 9.17), the primary card (21) is low and unmatched when compared to the next secondary (22). This card also is fed into the merge pocket. Unmatched primaries represent employees on the payroll who have no earnings for the current period. However, they must be included on the payroll register so they are not selected in this operation. The next comparison (Figure 9.18) shows that the secondary card (22) is low and unmatched. This card is selected from the file and fed into another pocket of the machine.

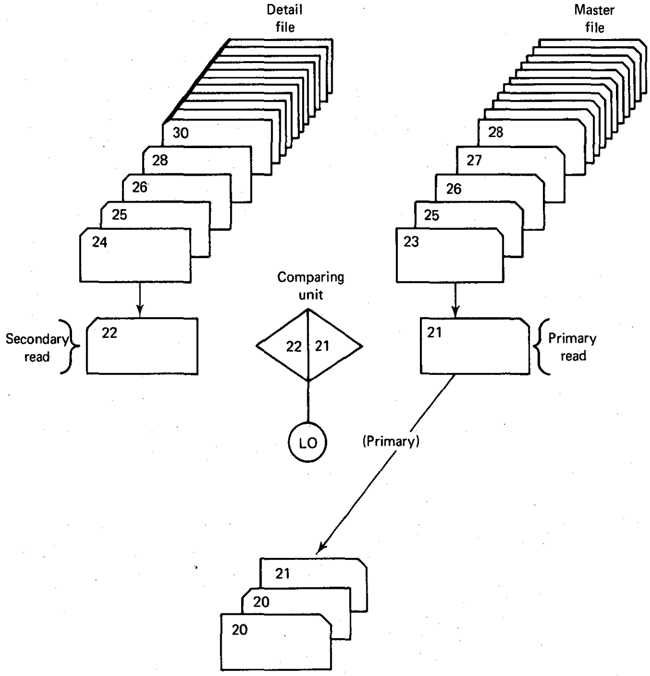


Figure 9.17 File merging with selection. Low primary.

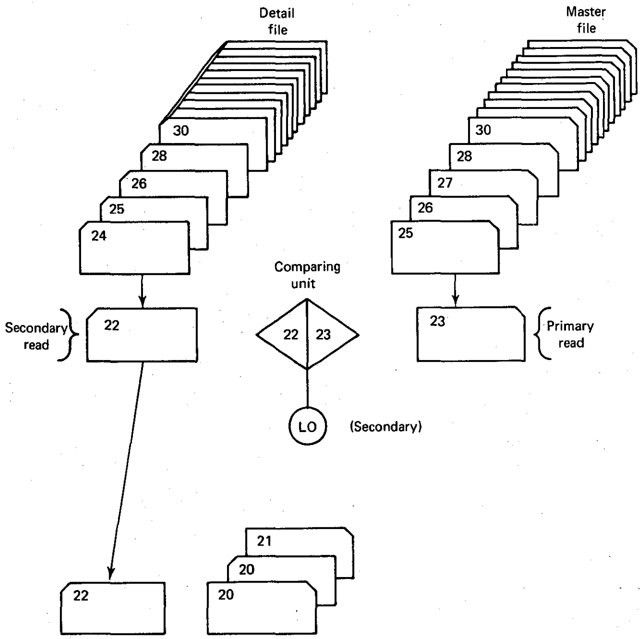


Figure 9.18 File merging with selection. Low secondary.

The merging and selection process continues automatically. Figure 9.19 shows that primary card 23 is also low and unmatched with no corresponding detail card and that one additional detail (24) has been selected. When merging has been completed, all unmatched secondaries will have been withdrawn from the file as exceptions to the regular procedure. Both files can be sequence checked while merging and selection is carried out.

In many cases it is also desirable to select master cards that have no matching details. For example, assume that a deck of sales transaction cards must be punched with a unit price before calculating a gross and net sales amount for each item. The item cards may be *priced* by first merging them behind master pricing cards and then transferring or *gangpunching* the price from each master to all following details. The sequenced control field for both files is item number.

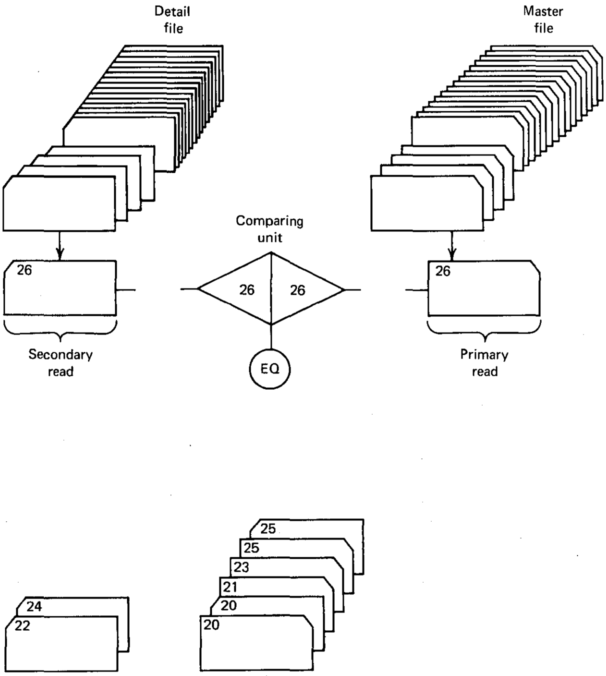


Figure 9.19 File merging with selection. Low secondary.

Figure 9.20 shows the pricing procedure in flow diagram form.

1. A batch of sales transaction cards is sorted to item number. The master file is maintained in this sequence.

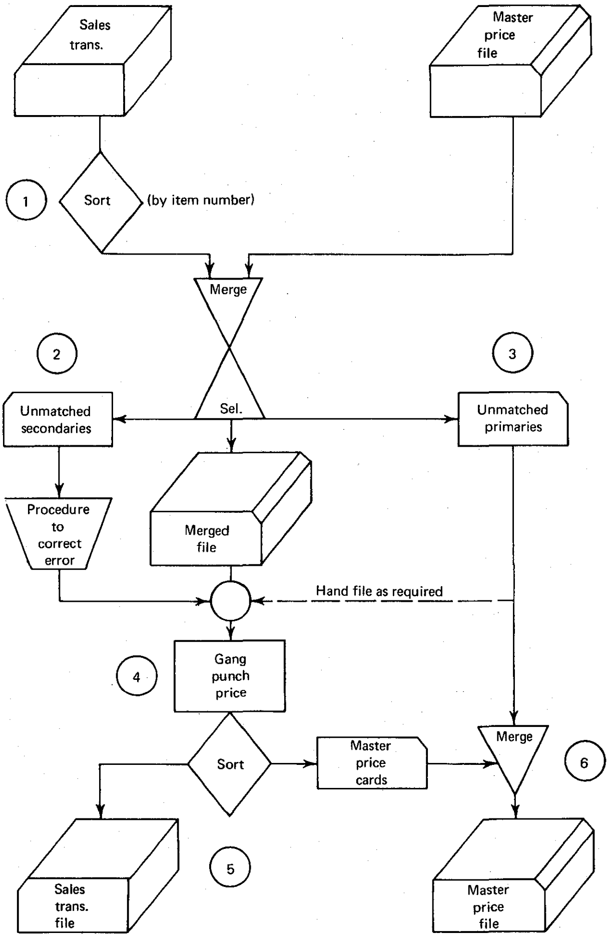


Figure 9.20 Pricing procedure. Flow diagram.

2. The two files are merged on the collator, selecting both unmatched primaries and secondaries. Unmatched secondaries are items that have no price. They could be new items in inventory for which no price has been established. Or, the selected cards might be punched with the wrong item number and therefore do not match the current price list. A machine operator or clerk must refer to the original document for correction or adjustment. When a price is established or an item number corrected, the matching cards are manually placed in the merged file in item number sequence.

3. Unmatched primaries are master price cards not used for this operation. The selected deck is put aside until gangpunching is completed.

4. The transactions are priced by transferring the unit price field from the master cards to following details.

5. The two files are sorted apart after gangpunching. Transaction cards are ready for further processing.

6. The selected master cards are returned to their file by merging without selection.

All operations are sequence checked by item number.

Merging with Equal Selection

The function of merging two files with selection can be utilized to update a master file by substitution. In this case, equal primaries are selected rather than low primaries or secondaries.

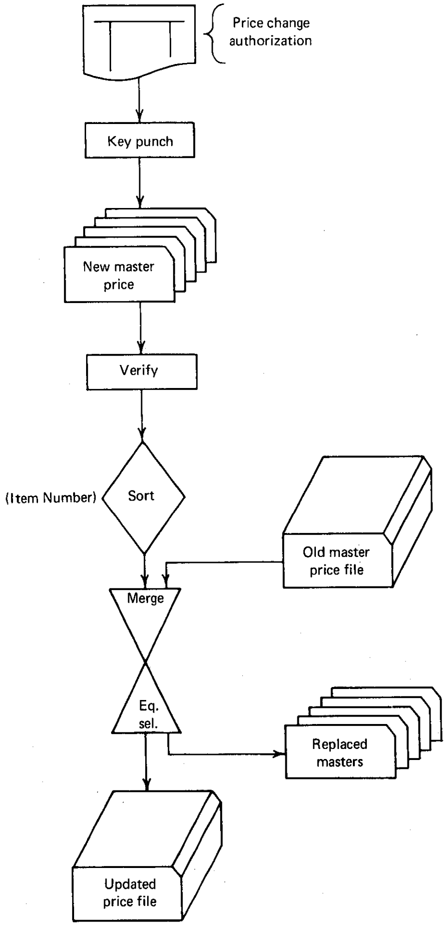


Figure 9.21 Updating master file by substitution.

For example, in the previous illustration a master pricing file was used to price detail transactions. When the price of any particular item changes, it is necessary to pull the old price card from the master file and substitute a new one. The operation is one of automatic merging with equal selection. A flow diagram is shown in Figure 9.21.

1. The new master price cards are punched from original records including item number, description, and unit price.

2. All cards are key-verified for accuracy. An error in unit price will cause an error in customer billing and consequently in the accounts receivable file. The error will be repeated every time the master card is used.

3. The new master price cards are sorted to item number.

4. New master price cards are merged into the master pricing file by item number. New cards are placed in the secondary; old masters in the primary feed.

5. As the secondaries are merged, equal primaries are selected. That is, the new masters replace the old matching masters in the file.

If required, any new masters without corresponding old cards can also be selected. However, this selection need not be made if there are item cards to be placed in the file representing entirely new items in inventory.

Both files are sequence checked for accuracy.

Other File-Handling Operations

The previous sections have dealt mainly with the type of file-handling operations that usually are carried out as processing before the updating of master records. However, there are other collating operations that can be performed to extract data directly from cards by various methods of manipulating the record files.

These automatic operations particularly show the unique advantages of the punched card as a "selectable" record. That is, cards can be automatically withdrawn from files at high speeds to locate specific classes or individual items of information, as required. Once withdrawn, the same cards can be returned to the file with equal facility by machine.

Some of these file-handling operations are described briefly in the following sections. The operations can all be performed on the 88 Collator.

Matching Two Files

The operation of matching one file against another is somewhat like comparing one detailed situation against another and making note of the exceptions. Instead of merging two files together in one pocket, matched files remain separated in the collator and are stacked in different pockets. Matching can be done between primary and secondary feeds, either by single cards or groups of cards regardless of the number of cards in a group.

Cards in either file that do *not* match can be selected. Thus, the two original files may be arranged as four files at the end of the matching operation: two matched files, selected primaries, and selected secondaries. All four groups remain in the sequence of the original files, that is, the ascending order of the control field.

Figure 9.22 is a schematic of a file matching operation. The master file contains the names and addresses of all current subscribers to a buying service. The file is in ascending numerical sequence by subscriber number.

Recently, all members were mailed descriptive brochures offering specially reduced prices on specific merchandise. Included with the brochure was a punched-card order form to be filled out, signed, and returned by the subscriber as an order.

The detail file in the illustration represents the returned orders, one card per subscriber. The detail file has been sorted to subscriber account number. The purpose of the file match is to determine which member subscribers have not yet returned their order forms. Those who have not responded are to be sent a second mailing reminding them that the offer expires shortly.

Master cards are placed in the primary feed of the collator, order cards in the secondary. The machine is set up to match the two files by subscriber number and to select unmatched primaries. Selected cards represent members who will receive the second mailing

Since the secondary cards could only have originated from the subscriber master records, it may not be necessary to wire secondary selection. However, there might be a special situation where a time limit for the merchandise offer has been set. Membership in the service might have expired for some subscribers. Consequently,

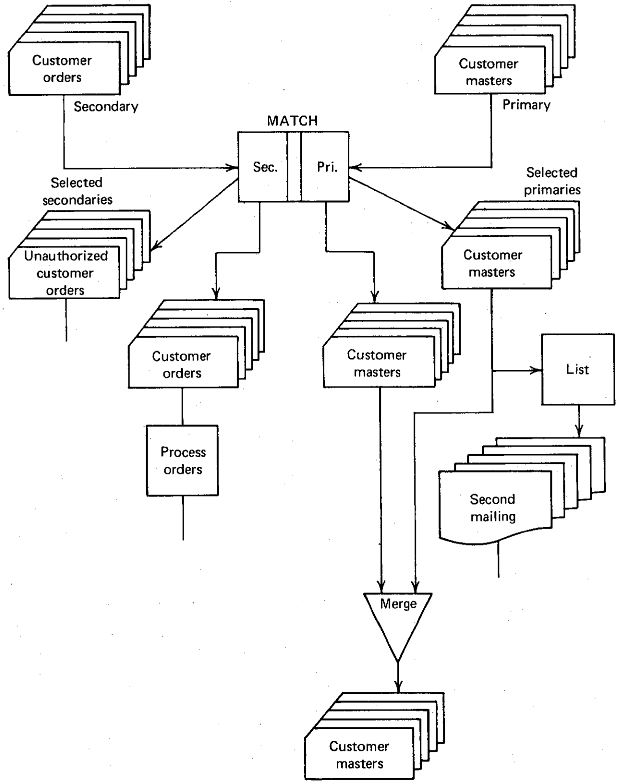


Figure 9.22 File matching.

their cards would have been removed from the master file. Selected secondaries can identify any former subscribers who are no longer entitled to the service.

After the second mailing is completed, the selected master cards are merged back into the file.

Editing As a check of punching accuracy, cards processed by the collator can be checked for errors of double punching and blank columns. Up to 22 columns in each feed, or up to 44 columns in one feed, can be checked. Because the 88 Collator is essentially a numerical machine, double-punch detection is automatic for every position read into a comparing unit. To detect unpunched columns, a switch for each position must be wired on the control panel. When a double-punch or a blank-column error is detected and the switch is wired, card feeding stops, a detect light is turned on, and another light indicates which feed contains the error card.

The error-detecting circuitry checks punching positions 0 through 9, but does not normally check 11 and 12 punches. However, complete detection can be specially wired on the control panel.

Double-punch and blank-column detection can be performed with other operations of merging and matching, or as a separate operation. When done separately, error cards can be selected into a separate pocket.

Comparing Two Fields in the Same Card

Two fields in the same card can be compared to determine whether they are equal or, if unequal, which of the two is lower or higher. Therefore, it is possible to select from a given file all cards in which field A is less than field B, field A is equal to field B, or field B is less than field A. The two fields are compared in a sequence unit and a low or equal comparison is selected.

A file need not be in any particular sequence, but both field A and field B must be punched in the same columns in all cards. A useful application of this feature is the selection of records from a given file where two amount or quantity fields are compared.

For example, a customer's total accounts receivable amount may be compared against a credit limit amount. Any accounts owing more than the limit can be selected from the file for appropriate follow-up and any required action.

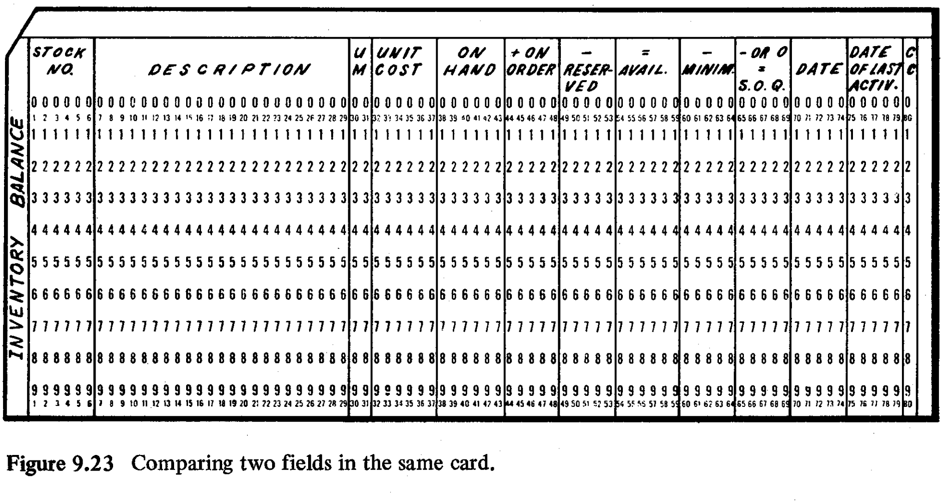


Figure 9.23 Comparing two fields in the same card.

Inventory status files often contain quantity fields for the availability of items. Figure 9.23 is a sample of how the card fields in such a record might be arranged. Note that two of the fields contain the quantities *available* and *minimum balance*. By comparing these two fields on the collator, any item with less than a minimum balance in stock can be quickly extracted from the file.

Selecting Cards by Either of Two Key Numbers

Cards punched with either of two key numbers can be pulled from a file in one run of the cards through the primary feed of the collator. The selected cards are stacked in two groups.

The two key numbers are punched in a *finder* card which is identified from all other cards in the file by a unique control punch. The finder card is placed at the front of the file by the operator before the run begins. By control panel wiring, the two key numbers are read into one side of the comparing unit where they are stored during the entire operation.

As cards feed, they are read at the primary read station and compared against the stored key numbers. An equal comparison in either field can be wired to select the corresponding cards. For this operation, the comparing unit is functionally split into two sections, one for each key number.

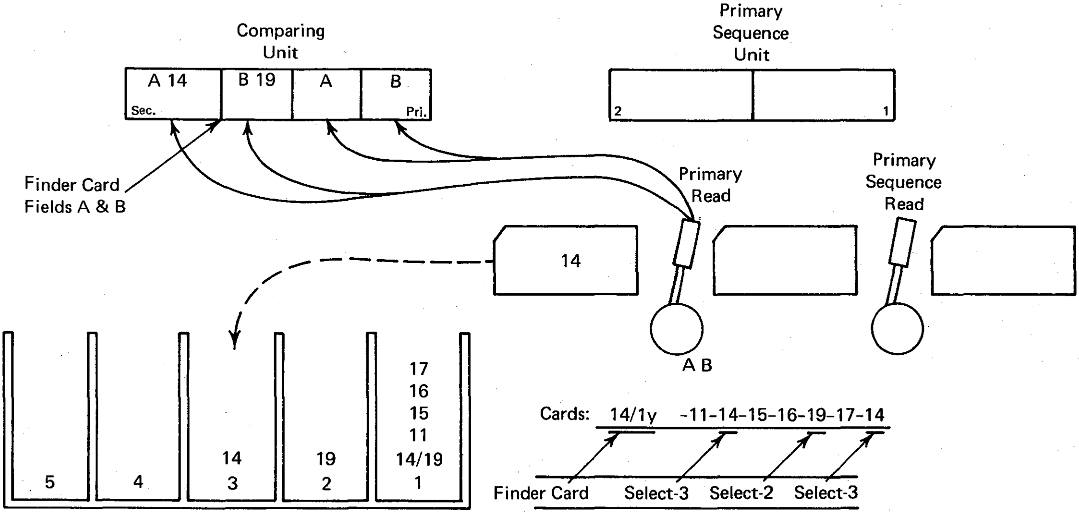


Figure 9.24 Collator planning chart.

The key numbers can be selected only in specified columns; that is, a search can be made for all cards containing the number

2645 in columns 22 through 25 and the number 1006 in columns 32 through 35. If the key numbers are punched in any other columns, those cards will not be selected. Figure 9.24 is a collator planning chart for this operation.

A number of other variations are possible. For example, cards may be selected from a file that are equal to one key number only, instead of two. Or cards may be selected that are either higher or lower than a given key number. Finally, all cards may be pulled from a file that are between two numbers representing maximum and minimum limits.

Key numbers are stored in the comparing unit of the machine by reading finder cards before a run begins. Each finder card must be identified by a unique control punch to distinguish it from all other cards in the file, like an 11 punch in column 20 when all other cards have only digit punches in that column. The key number is always punched in known fields, both in the finder cards and in the cards of the file to be searched.

The type of operation to be performed is set up beforehand by proper control panel wiring.

Selecting Zero-Balance Cards

Cards can be selected from a file that are punched with zeros in any particular field. These *zero-balance* cards may be selected from either the primary or secondary feed, or from both feeds simultaneously. By placing half of the file in the primary feed and half in the secondary feed, processing time is reduced. When both feeds are used, a maximum of 33 columns per card can be checked for a zero balance. If all cards are fed through the primary, up to 66 columns can be checked in one run.

Zero balances are detected by entering the field to be checked into one side of the comparing unit, and *emitted zeros* into the other side. That is, by control panel wiring, impulses timed as zeros can be generated internally by the machine. These emitted impulses can also be used to control machine functions during certain operations in the same way as impulses obtained by reading cards. In this case, a comparison between emitted zeros and zeros read from cards can be recognized as equal. The equal comparison causes automatic selection of the corresponding card.

Blank columns are not recognized as zeros. Therefore, a field must be fully punched to be selected.

As an example of how this feature might be used in an actual data processing application, refer to Figure 9.25, an illustration for an inventory stock status report. It may be assumed that the information listed on the report is also punched in cards, one card per line.

INVENTORY STOCK STATUS REPORT

Date____________

Stock No.	Descrip.	Old Bal.	Receipts	Issues	Adjust		On Hand	On Order	Reserved	Available	Min. Bal.	Below	Standard Order Quantity
		+	+	−	±	cr	=	+	−	=	−	0	
1234	Bolt	150	25	40	10		145	30		175	100		
1235	Nut	300	50	150	25	cr	175	50		225	150		
1236	Screw	500	100	300	100	cr	200	100		300	400	*	1000
1237	Nail	400		350	50	cr		200	100	100	200	*	8000

Figure 9.25 Inventory stock status report.

In control inventory applications it is often necessary to quickly identify items with zero balances: items out of stock or items with zero availability. Any such critical items can be readily located in a file by searching for zero balances in the on hand and availability fields. Once located, proper action can be taken to replenish stock to meet estimated requirements.

Checking the Filing of Details Behind Master Cards

A file of punched cards is often made up of master cards with corresponding detail cards. Such a file can be checked automatically to be sure that each detail follows the proper master. The check may be particulary necessary when manual filing of cards is part of the procedure of assembling the file.

When the cards are in some ascending sequence by a control field, the operation can be done by a normal sequence-checking operation. However, if the groups are not in any particular order, they can still be checked if the collator is properly set up by control panel wiring.

The master cards must be uniquely identified by an 11 or X punch in a specified column. The control number must be punched in the same field in both master and detail cards.

Cards are placed in the primary feed of the collator. The master card control field is entered into one side of the comparing unit and held until the next master card is fed. That is, when the distinctive X punch is read by the machine, the comparing unit is cleared and the new master card control number is read in.

Each detail card number is read into the other side of the comparing unit. An equal comparison indicates that the detail follows the matching master card. An unequal impulse (either high or low) indicates that the detail card is misfiled. The unequal impulse can be wired to select unmatched cards into a separate pocket. They can then be refiled properly.

A number of variations on this type of operation are possible. The presence or absence of a specific X-punched card within a group can cause a special card to be inserted after that group of cards in a file. "Groups" of cards are considered to be those which are all punched with the same control number in a specified field.

For example, in a billing operation, the inserted card can be a special discount, terms allowed, or description card to explain the presence of symbols on selected customer invoices. It is assumed that the X-punched card within the group signals the need for additional information on the invoice. The special card is inserted behind a group of cards that all contain the same customer or invoice number. However, the inserted card need not be punched with the control number of the group.

Conversely, the last card of a group can be pulled from the file only if it is punched with a specific X punch. This operation will remove those cards previously filed in the preceding operation.

Note that these operations cannot be done on the sorter because selection is made both by control number change and signal card.

The ability of the machine to recognize the first card of a group by a change in control number can be used to select that card, leaving all other cards of the group in the file. Thus, master cards can be pulled even when the file is in no particular order and when the master cards are not punched with a distinctive identification.

Single cards can be identified and removed from a file which supposedly contains only multiple groups. The single cards are selected, leaving the multiple groups intact in a separate pocket of the collator.

In some accounting machine operations, the last card of each control group must be an X-punched card. For example, in a payroll operation it may be necessary that the last card for each employee be a summary earnings card punched with a specific X code. This can be checked on the collator before the cards are run on the accounting machine.

In this operation, the last card of a group is selected if it is *not* punched with the required X code. The selected cards indicate to the operator which groups must be adjusted before the cards are processed further.

Questions and Exercises

The following exercises refer to the stock room procedure started in Chapter 6.

1. Consider setting up a master punched-card file for all items carried in stock, one card for each item. The card may serve as a pricing record and an inventory control record. What fields should the master card contain?
2. How is the master file to be originated? Maintained? Controlled? Sequenced?
3. Assume that requisitions will be accumulated for a given period and then sorted into the same sequence as the master pricing file. Merge the requisitions as secondaries behind the master file as primaries. Select unmatched primaries and secondaries. Explain the origin of any unmatched requisitions and unmatched master cards.
4. List the specifications for each master card field and design a card form as a pricing record. The transfer of price from one record to another will be explained in the next section.
5. Include controls for these new aspects of the procedure.
6. Present your proposal in class.

DATA TRANSFER BY PUNCHED HOLES 10

Punched card records have another unique advantage over many other types of documents. Information in cards can be automatically copied from one machine-readable record to another by the direct duplication or *reproduction* of punched holes. The copying process can be varied in a number of ways, from the direct duplication of a complete file, card by card, to the automatic transfer of selected data from one card to one or more *gangpunched* cards.

This chapter describes some of the basic methods of punched card information transfer. The processing is done by special equipment designed for this purpose, notably the IBM 519 Document-Originating Machine.

Information can be transferred in three different ways:

1. *Gangpunching*—the punching of information from a master card into a following group of detail cards. Masters and details are identified to the machine by unique X punches.
2. *Reproducing*—the card-by-card duplication of punched holes from one document to another. All or any part of the data from a card can be punched into a duplicate card.
3. *Summary Punching*—totals accumulated by an accounting machine can be punched into a *summary* card. For this operation, the 519 must be connected by cable to the accounting machine equipped with a special feature by the manufacturer.

Two automatic machine checks can also be used with these information transfer operations:

1. *Comparing*. While reproducing or gangpunching, the machine can compare the duplicated hole pattern with that of the original card. Any errors are indicated automatically for the operator's attention.

2. *Double-Punch and Blank-Column Detection*. The presence of two or more holes in a column can be detected. The absence of any holes can also be detected. The use of this feature will be more fully explained with the operation of mark sensing.

Additional special operations can be performed by the 519:

1. *Mark Sensing*. Information recorded by pencil marks on 80-column cards can be punched into the same cards when this special feature is installed.

2. *End Printing*. Up to eight digits of information can be printed on the column 1 end of the card. The printing is in a specially designed large type positioned to be read when the card is filed vertically with the column 1 end up. For example, end printing is used to originate punched clock cards for employee attendance registration.

Gangpunching

Figure 10.1 is a schematic of the card feeds on the 519 Document-Originating Machine. Like the collator, the machine is designed to feed two files of cards simultaneously. Therefore, the unit is equipped with two card transport mechanisms with corresponding sets of sensing brushes. The machine may also be considered to have two separate functional units that can operate independently on a single file, as required. These units are labeled *read* and *punch* in the illustration.

The punch unit will be described first for the operation of gangpunching.

As cards move from the hopper, they pass under a rotating mechanism equipped with a set of 80 punches. The punches are positioned above the card with 80 matching rectangular dies below. Movement of each card through the machine is timed precisely so that any of the 960 possible punching positions can be perforated by extending a punch downward through the card into the die.

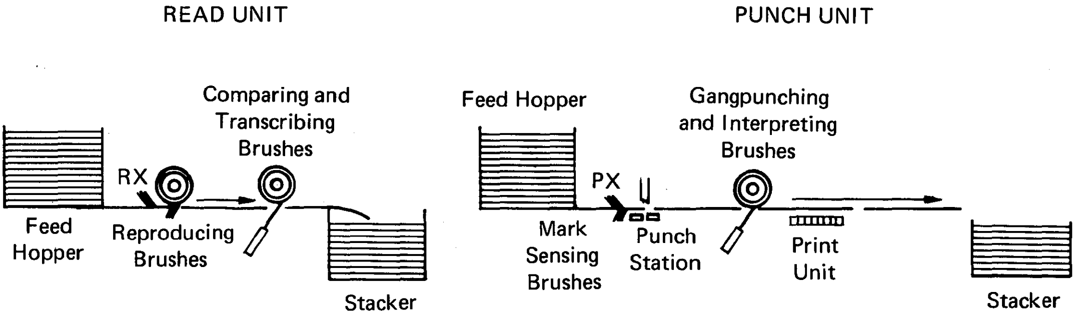


Figure 10.1 Schematic of card feeds, IBM 519 Document Originating Machine.

Punches and dies are in exactly the same arrangement as a single row of hole positions in a card.

Since a card passes the punching station row by row, each card is punched in the same fashion, that is, all 12 holes are perforated first, then the 11 holes, then 0, 1, 2, and so on to the 9 row.

However, to actually cut holes, the punching mechanism must be actuated at the proper time in the feeding cycle. This is done by sensing the preceding card at a reading station. Here, cards pass between a set of 80 gangpunching brushes and a metal roller where any card field can be read. Selection of the columns or fields to be read and punched is part of the process of setting up the machine before operation begins.

The card being read moves under the gangpunching brushes exactly synchronized with the following card at the punching station. The 12 row of this card is under the brushes at the same time the 12 row of the following card is over the punching dies. All the following rows also pass the reading and punching stations at the same time. Thus, to actuate the punching mechanism for any particular column, it is only necessary to connect or wire the punch for that column to the corresponding brush at the reading station. The electrical impulse obtained by sensing a hole is used to copy or duplicate that hole from one card to the next.

The 519 is equipped with a control panel. By panel wiring, any gangpunching brush can be connected to the corresponding punching die. When the control panel is inserted and the proper control switches turned on, the machine will perform the operation of gangpunching in the punched feed.

For example, assume that a procedure requires a file or deck of cards to be dated for accounting purposes. The date will be punched as a six-digit number in columns 1 through 6.

Before operation begins, the control panel of the 519 must be wired to connect gangpunching brushes 1 through 6 to punching positions 1 through 6. Next, the first card of the deck is manually keypunched with the correct numerical representation of the date, e.g., 013172 represents January 31, 1972.

The deck of cards is then placed in the hopper of the punch unit. When the machine is started by the operator, the first card is transported past the punching station in one feeding cycle. Since no card has yet reached the gangpunching brushes, the card passes the punching dies already punched.

During the next feeding cycle, the following card (blank in columns 1 through 6) passes the punching station while the first card passes the gangpunching brushes. Since brushes 1 through 6 are connected to punches 1 through 6 via the control panel, all holes are gangpunched from the first card to the second.

On the next feeding cycle, the third card passes the punching station while the second card passes under the gangpunching brushes. The punching in columns 1 through 6 is gangpunched from the second card to the third. The process is repeated until all cards are punched. Thus, the information from the first card is automatically copied into all succeeding cards of the file.

Accuracy of the entire gangpunching operation can be verified by sight checking the last card in the file with the first card. The punching in columns 1 through 6 should be the same (Figure 10.2).

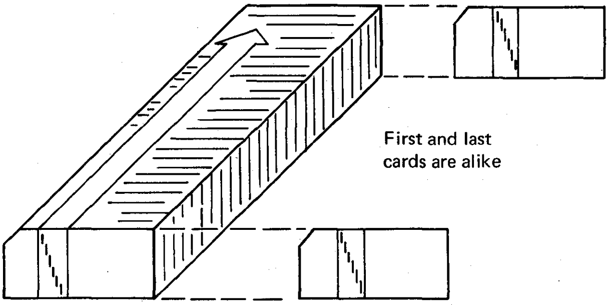


Figure 10.2 Schematic. Gangpunching.

Notice that the machine is set up for this operation in such a way that *any* holes in columns 1 through 6 are copied from the card at the gangpunching brushes back to the corresponding columns of the card at the punching station. If, by error, there are any holes in columns 1 through 6 of the card *other* than the date 013172, they will be picked up and also copied into all following cards. If this happens, double punching can occur in one or more of the columns and the file will be damaged.

To prevent this type of error, double-punch detection for the gangpunched columns can be wired on the control panel. If any double punching occurs, feeding halts and an indicator light comes on to signal an error. The operator can disconnect the wiring between the gangpunching brushes and the punching station by removing the control panel. The machine is then cleared of cards. The operation can be restarted by punching the first card of the remaining deck with the proper date.

Interspersed Gangpunching: X-Punched Masters

Interspersed gangpunching is the process of punching data from master cards into matching detail cards. The master cards must have been previously collated or hand-filed ahead of the details in sequence by an identification field; for example, man number, part number, or account number.

The purpose of the operation is to transfer some specific field of information from each master card into the following detail cards. The gangpunched data can be used in subsequent steps of processing that include calculating and printing. The data is variable from one master card to another. Common examples include price, size, description, rate, and name.

The operation is similar to the process of straight gangpunching—with one important difference. The read-back from the gangpunching brushes to the punching unit must be interrupted each time a master card is at the punching station. Otherwise, data from the last detail of a group will be double punched into the next master.

To interrupt punching for master cards only, the machine must sense which cards are master cards and which are detail cards. As previously stated, master cards are normally identified with a distinctive X punch in a specified column.

A special reading station is located just ahead of the punching station of the 519 to sense X punches only (Figure 10.3). Movable Punch X-brushes are mounted on a notched bar above the card path through the punch unit. A brush may be set on any column position along the bar and clamped in place with a setscrew. Six Punch X-brushes are standard equipment with the machine.

Each Punch X-brush has a corresponding exit hub on the control panel. When an X punch is sensed, the resulting electrical im-

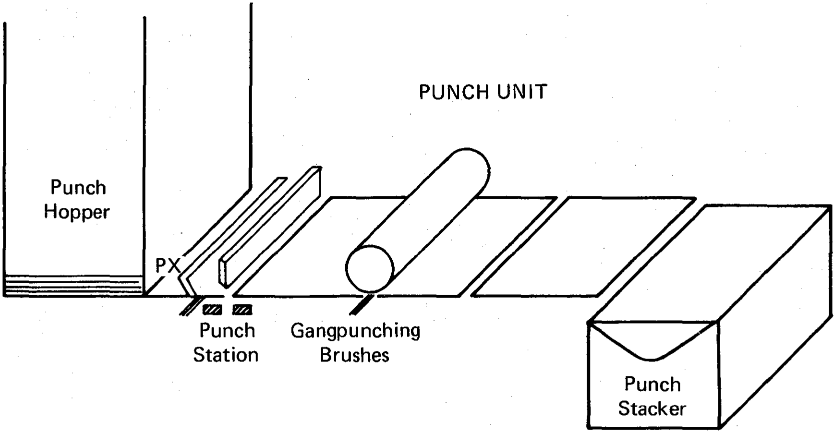


Figure 10.3 Punch unit. IBM 519.

pulse can be wired to cause suspension of all punching during the following feeding cycle. That is, the cycle when an X-punched master card is under the punching station.

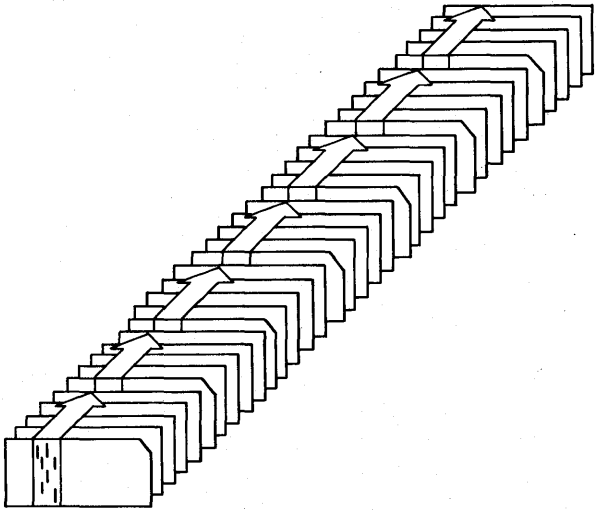


Figure 10.4 Interspersed gangpunching.

Figure 10.4 is a schematic of interspersed gangpunching. Masters with varying numbers of matching details are fed through the punch unit. The card field in columns 25 through 30 is wired to gangpunch.

Interspersed Gangpunching: X-Punched Details

Although the above procedure is operationally correct, there is a distinct advantage in using a unique X punch in all details rather than an X punch in masters. This is to prevent damage to a file by double punching.

For example, if an X punch in a master card is *not* sensed, punching will not be suspended and double punching will result. To prevent this situation, the 519 can also be controlled to suspend punching for all No-X cards.

In this case, the Punch X-brush is set on the column containing the *detail* X. Whenever an X punch is *not* sensed (master cards), punching is suspended. With this system, failure to read an X punch will result only in a blank detail with no damage to the file.

All gangpunching can be checked for accuracy by comparing in the read unit of the machine as follows.

Checking Interspersed Gangpunching

The read unit of the 519 is equipped with two sets of 80 brushes as shown in Figure 10.5. By control panel wiring, the punching in a card passing the comparing brushes can be compared with the following card as it passes the reproducing brushes.

The field to be compared is wired from each set of brushes to opposite entries of a comparing unit. These connections can also

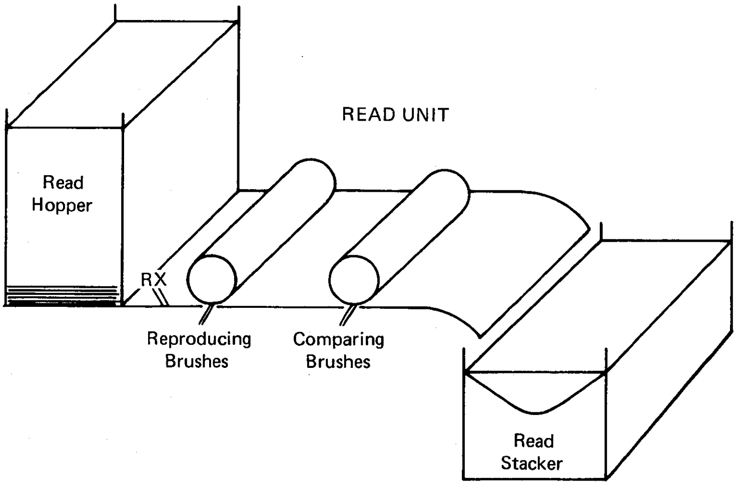
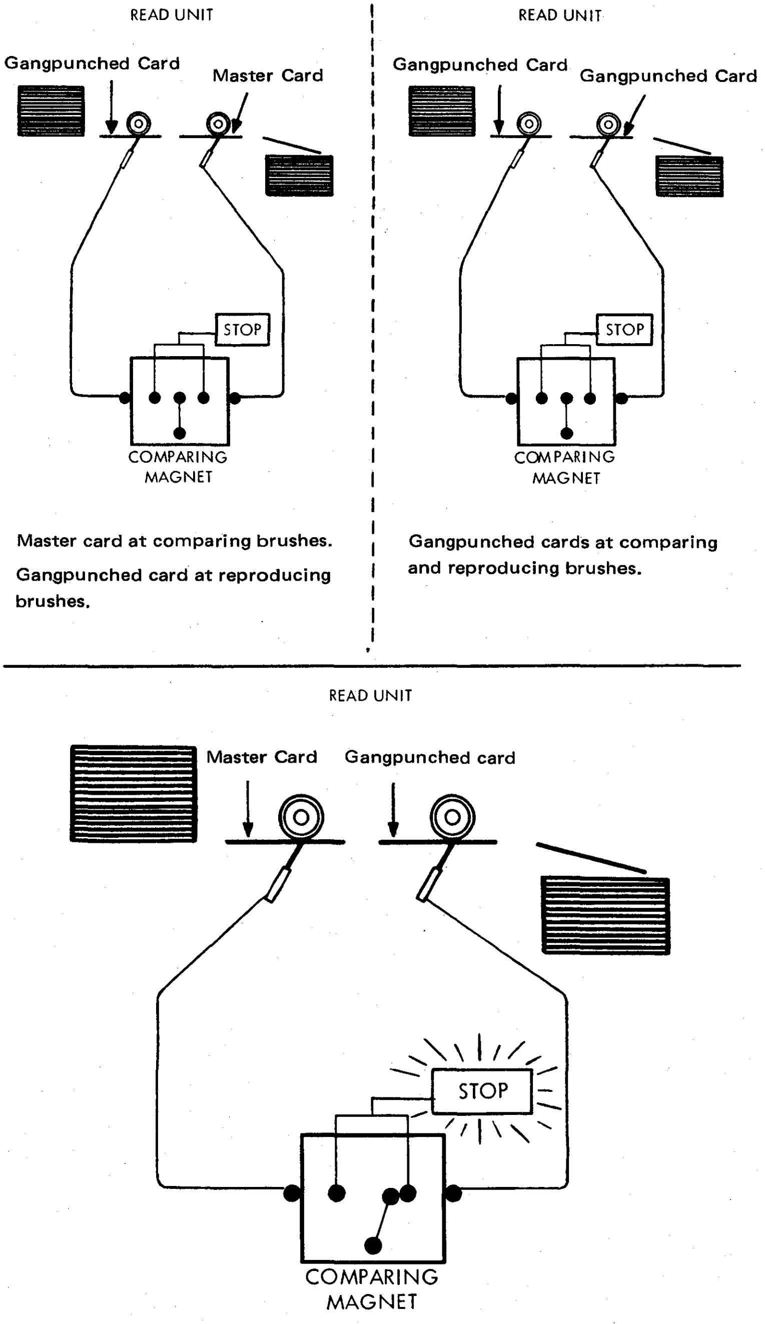


Figure 10.5 Read unit. IBM 519.



Master card at comparing brushes.

Gangpunched card at reproducing brushes.

Gangpunched cards at comparing and reproducing brushes.

Gangpunched card from previous group at comparing brushes.

New master card at reproducing brushes.

Figure 10.6 Gangpunch compare. Master to detail, detail to detail, detail to master.

be made on the panel. Any difference in punching between the two compared fields is sensed by the unit. The resulting electrical impulse signals the machine to stop. At the same time, an indicating mechanism displays the positions of the columns that do not compare. The operator can remove the cards to investigate the cause of the trouble.

The gangpunch compare operation requires that either master or detail cards must be punched with a distinguishing X. Like the punch unit, the read unit has special Read X-brushes for this purpose. Refer again to Figure 10.5. The brushes are movable and can be set to read any specified column. When an X punch is sensed, comparison between the reproducing and comparing brushes is interrupted for the next feed cycle. This prevents a comparison between the last detail of a group and the following master. This situation is shown schematically in Figure 10.6. Comparison may be suspended for either an X or a No-X reading.

To check interspersed gangpunching, the file is taken from the punch unit stacker and placed in the hopper of the read unit. Each feed operates independently and one feed will operate while the other is empty.

Offset Gangpunching

In the previous illustrations of gangpunching, it was assumed that card fields are always punched into the same columns of all cards in a file. That is, a field in a master card is punched into the corresponding columns of each following detail of a group.

Occasionally, a gangpunching operation involves punching from a master card into different columns of the detail. This operation is called *interspersed gangpunching.* For example, the information in columns 50 through 54 of the master cards can be punched into columns 3 through 7 of the detail cards. This means that *field selection* is necessary. Whenever a master card passes the gangpunching brushes, columns 50 through 54 must be punched into columns 3 through 7 of the following detail. Whenever a detail card passes the gangpunching brushes, columns 3 through 7 must be read and punched into the following card—if that card is a detail. When a master card follows a detail, punching must be suspended as in normal interspersed gangpunching to prevent double punching the master card. The field selection between columns 50 through 54 and columns 3 through 7 is made at the gangpunching brushes, since all *punching* is done in columns 3 through 7.

Columns 50 through 54 are only read; they are never punched.

As in normal interspersed gangpunching, master or details must be identified to the machine by a distinguishing X punch. In the punch unit, the X is also read by a Punch X-brush. The resulting impulse stops punching action between a detail and master and, one cycle later, causes selection of columns 50 through 54 as an X field. Figure 10.7 is a schematic of offset interspersed gangpunching.

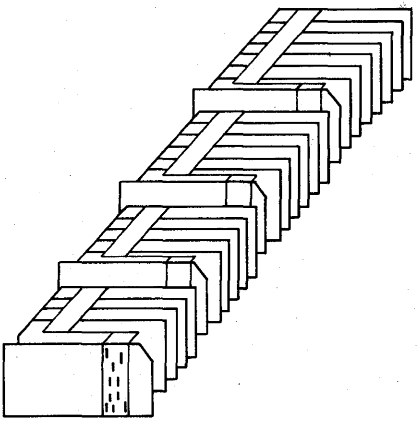


Figure 10.7 Interspersed gangpunching.

Comparing Offset Gangpunching

When offset gangpunching is compared, field selection is used similar to that required when the cards were punched. The selection must be made to read the proper field from the card at the comparing brushes. When a master card is at the comparing brushes, the master field must be read for comparison with the detail card at the reproducing brushes. When a detail card is at the comparing brushes, the detail field must be read for comparison with the following detail at the reproducing brushes. The detail field is always read at the reproducing brushes, and comparison is suspended when a master card is at the reproducing brushes.

The detail field from the reproducing brushes is entered into one side of the comparing unit, and the master or detail field from the comparing brushes is entered into the other side of the comparing unit.

Selecting the fields read at the comparing brushes requires the use of a *selector*, a kind of switch that can be wired from the control panel. The selector is controlled by the identifying X in the cards. As in the punch unit, the X reading must be delayed one cycle. At that time the X punched card is at the comparing brushes and the selector must be controlled.

Whenever gangpunching is compared, it is also good operating practice to compare the identification fields at the same time. The comparison checks the filing previously done on the collator or by hand. If a master is filed ahead of the wrong details, or if a detail is misfiled, the comparison check indicates the error in the same way that incorrect gangpunching is indicated.

Selectors

Selectors can be installed as special features in the 519 and other card machines. Each selector is a two-way switch that can be controlled automatically to select a connection over one of two alternate paths.

Most common switches are operated manually. That is, a switch is positioned in one direction or the other by moving a handle, pushing a button, or throwing a lever. The selectors on card machines are operated automatically by impulsing a pickup hub.

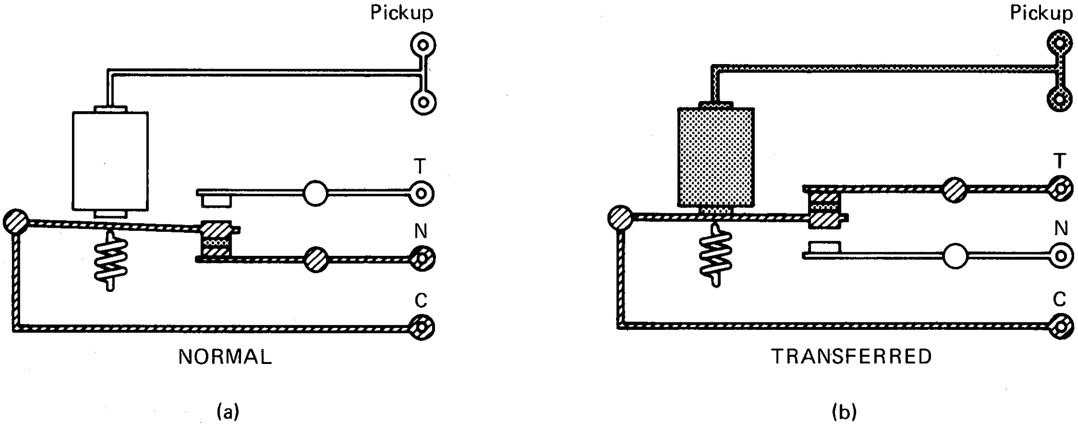


Figure 10.8 Selector. Normal and transferred.

Figure 10.8a is a diagram of one position of a selector set at normal. An electrical connection is established between C, or common, and N, or normal. The switch is held in contact at the normal position by spring tension.

An electromagnet is positioned in relation to the switch as shown. Assume now that the magnet is impulsed through the pickup hub (Figure 10.8b). The resulting magnetic attraction lifts the switch against the retention of the spring and makes contact with T, or the transferred side of the selector. In this position, a connection is made between C and T. The normal hub is disconnected when the transferred hub is connected, and vice versa.

Figure 10.9 is a diagram of a selector as it appears on a card machine control panel. The small circles represent holes in the panel where plugwires can be inserted. A selector has two positions, all controlled in unison from a single pickup hub.

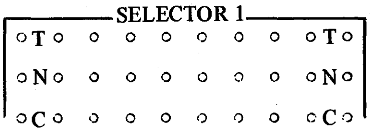


Figure 10.9 Selector. Control panel diagram.

If a card field is wired from reading brushes into the normal side of the selector, the resulting impulses are always available at the C or common hubs.

If a second card field is wired into the T, or transferred side of the selector, the resulting impulses will be available at C *only* when the selector is transferred. Transfer is effected by impulsing the pickup hub with an X reading from a Read X- or Punch X-brush. This action of the selector makes it possible to read offset card fields for the operation of interspersed gangpunching.

Reproducing

To duplicate or *reproduce* a file of cards, the read and punch units of the 519 Document Originating Machine operate together. Original source cards are placed in the feed unit and cards to be punched are placed in the punch unit. Cards are fed from both hoppers simultaneously, one at a time.

Feeding cycles are timed so that a source card passes the reproducing brushes, row by row, just as a card to be punched passes the punching station. The entire source card or any selected fields and columns can be reproduced, hole by hole, into the card at the punching station.

By control panel wiring, the fields in the reproduced card can be punched in any desired arrangement, regardless of the way the data appears in the original cards. For example, columns 1 through 20 can be duplicated into columns 61 through 80, or vice versa. All 80 columns can be reproduced, column for column, as required.

Comparison of the original and duplicate information can be done one cycle later on the same run of the cards through the machine. When the source card is passing the comparing brushes, the reproduced card is passing the gangpunching brushes. At that time, the reproduced card is compared with the source card, using the comparing unit. Any disagreement stops the machine. Columns that do not compare equally are indicated to the operator for correction. Figure 10.10 is a schematic of the reproducing operation.

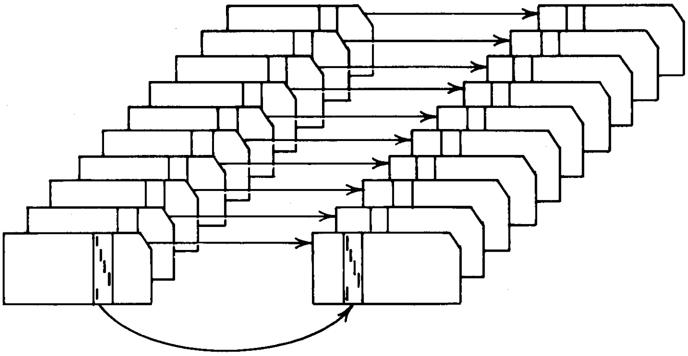


Figure 10.10 Reproducing schematic.

In a data processing procedure, a great deal of flexibility is provided by the ability to duplicate files automatically in any arrangement. For example, any selected information can be reproduced from a file for processing while the source file remains in its original sequence available for reference.

Report processing can often be expedited by creating a duplicate file which can be sorted to another sequence while the first file is being run on an accounting machine.

Two files can be matched, card for card, on the collator. Then, selected information can be reproduced from one file into another. At the same time the matching identifying fields can be compared to check matching.

The 519 is also designed so that gangpunching can be done in the punch unit while information is being reproduced into the same cards. Thus, a duplicate deck can be given an identifying punch or mark to label it as *duplicate* in all subsequent processing.

Selective Reproducing

It is often desirable to reproduce only certain cards from a file, without disturbing their arrangement. Cards to be reproduced, or those *not* to be reproduced, are identified by a unique X punch. The operation is like normal reproducing except that punching is suppressed for unwanted cards.

During the operation, the two feeds continue to operate in unison. Cards not reproduced are completely blank as they are fed into the punch unit stacker. The machine can be equipped with an *offset stacking device* that leaves the ends of the blank cards offset from the reproduced cards. If there are too many blanks to be removed by hand, blanks can be separated from the deck on the card sorter.

Figure 10.11 is a schematic of selective reproducing.

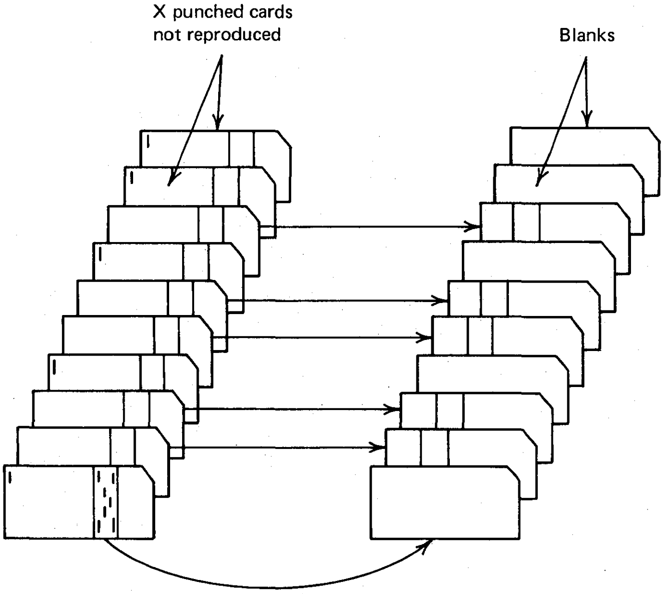


Figure 10.11 Selective reproducing.

Combined Reproducing and Gangpunching

For all combined reproducing and gangpunching operations, the read and punch units operate together. If gangpunching is to be performed from a single master card, a blank card should precede the cards in the read unit. This is necessary because the master gangpunch card in the punch unit should be one card cycle in advance of the first source card in the read unit to avoid reproducing into the gangpunch master card. The reproduced data can be compared in this same operation. Figure 10.12 is a schematic of this operation.

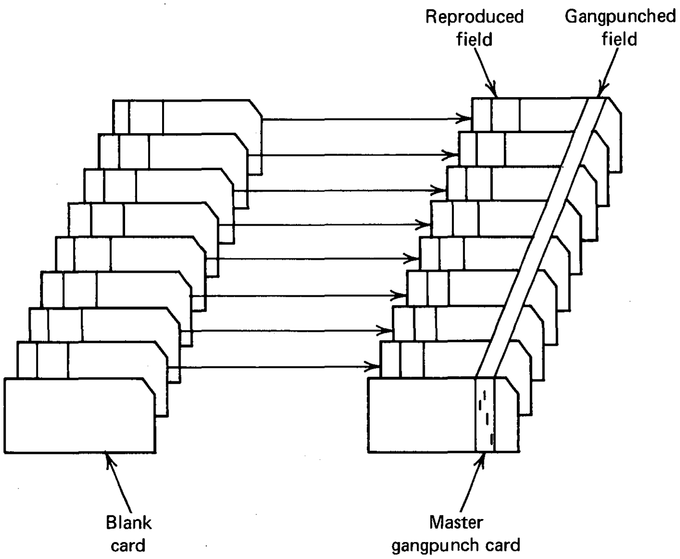


Figure 10.12 Combined reproducing and gangpunching.

Interspersed gangpunching can also be performed while reproducing. To prepare for this operation, two previous collating operations must be completed:

1. The source cards and the cards into which information is to be reproduced are matched card for card. Both decks must therefore be in sequence by some identifying field.

2. The cards to be reproduced are merged behind X-punched master cards by the same identifying field. If there are relatively few masters, they might be hand filed.

The masters and the merged details are placed in the punch unit. The source cards are placed in the read unit. By control panel wiring, the 519 can be set up to:

1. Suspend punching whenever the X (or No-X) master card is at the punching station.

2. Read from the gangpunching brushes to gangpunch the specified field into the details.

3. Feed the master card in the punch unit one cycle *without* feeding a source card. This is necessary in order to keep the two files exactly matched, one source card for each detail.

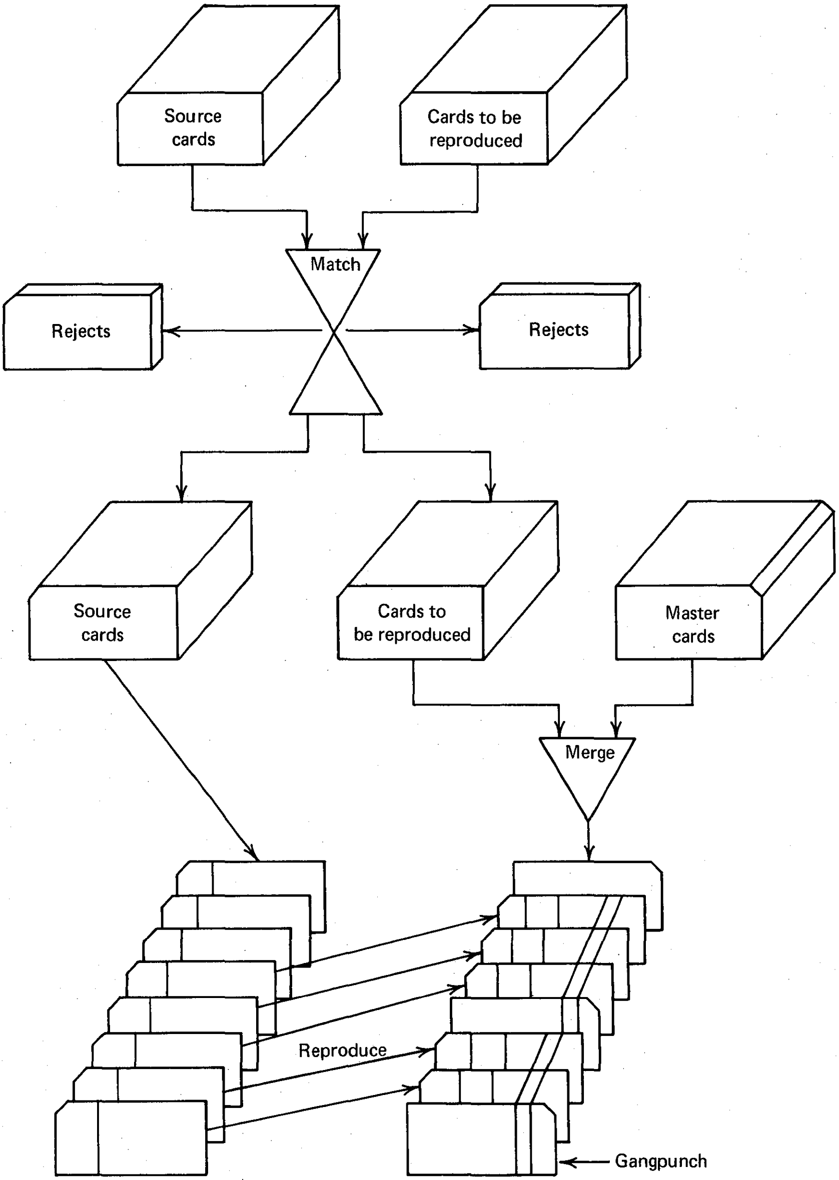


Figure 10.13 Reproducing and interspersed gangpunching.

4. Reproduce from the source cards to the detail cards.

Both gangpunching and reproducing can be compared for accuracy. Figure 10.13 is a schematic of this operation.

End Printing

The 519 can also be equipped with an end printing unit that will print numerical digits on the end of an 80-column card. The unit has eight print wheels, each with the digits 0 through 9 and one blank space. The location of the print unit in the machine was shown in Figure 10.1.

Numerical information to be printed can be read from the punching in the same card or from a card at the comparing brushes in the read unit. Figure 10.14 shows that information can be printed on either of two lines. The top of the first line is $\frac{5}{16}$ inch from the end of the card; the top of the second line is $\frac{5}{8}$ inch from the end of the card. If printing is to be done on two lines on the same card, two runs of the cards through the machine are required.

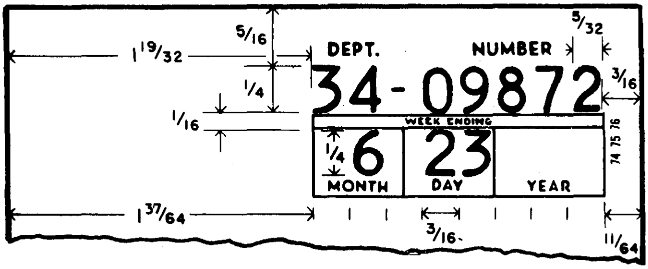


Figure 10.14 End printing.

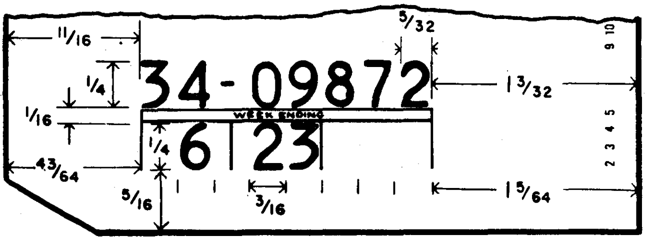


Figure 10.15 Inverted end printing.

The position of the printing line is selected by latching the printing unit in one of two notches in a rail on which the unit slides. Selection is done by the operator before operation begins. When not in use, the unit is disengaged by sliding it to a third notch in the rail.

The unit also can be specified for inverted end printing, sometimes required in the preparation of merchandise tags and preprinted stub cards. Samples of this type of printing are shown in Figure 10.15. For this application, the print wheels are permanently installed in the inverted position.

Mark Sensing

Original information on an 80-column card document can be transcribed directly into punched holes by a process called *mark sensing.* The information is first marked with a pencil in predetermined spaces on one surface of the card. From there it is *sensed* by the 519 Document-Originating Machine and automatically punched into the specified columns. The marked information can be punched into the same card or into a duplicate card.

For marking, a card is divided into 27 vertical columns. One mark-sensing column occupies the same space as three punched-hole columns. Each mark-sensing column is subdivided into 12 vertical spaces corresponding to the 12 punching positions in the card (Figure 10.16).

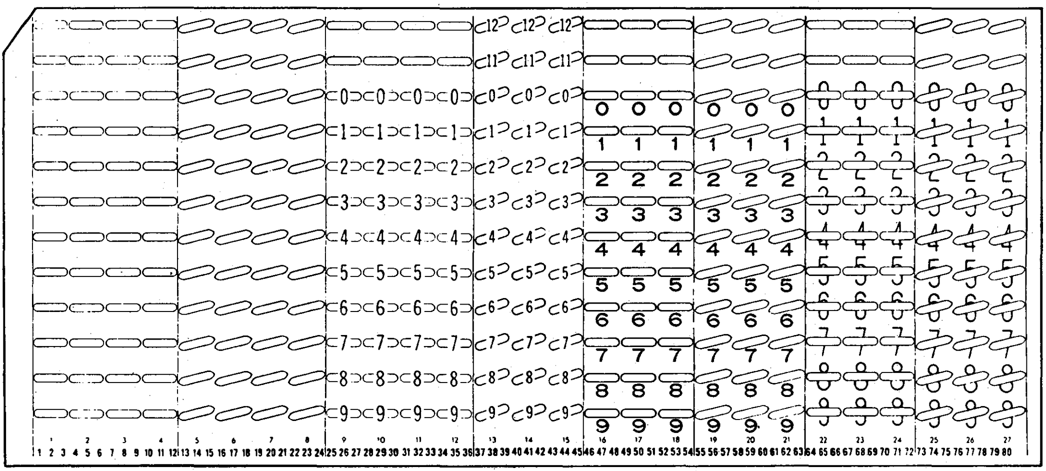


Figure 10.16 Column arrangement and eight styles of mark sensing.

The 519 can be equipped with a special device to sense the marks and punch corresponding holes, as wired on the control panel. *Mark-sensing brushes* can be fitted to either the punch or read units. The position of the brushes in the punch unit was shown in Figure 10.1. Normally, marked cards are fed through the punch unit, where the marks are sensed and then punched into the same card. However, when the read unit is equipped with mark-sensing brushes, marked cards may be fed through the read unit and punched into a duplicate card in the punch unit.

The sensing of the marks is based upon the principle that a pencil mark on paper can conduct electricity. The mark must be made with a lead of high graphite content. Best results are obtained when cards have been clearly and carefully marked. Because the marks must conduct electricity, only a special *electrographic* pencil should be used. Ordinary pencil lead, colored pencils, or crayons do not contain enough conductive material. A dense, black, narrow mark from a sharp pencil is always a better conductor than a broad light mark. The dense mark is much more likely to contain a continuous unbroken deposit of graphite on the surface of the paper. For best results, the card should be placed on a hard, smooth surface so that marks are not indented into the paper.

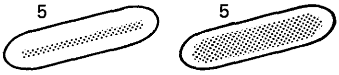


Figure 10.17 Mark-sensing marks.

The left mark in Figure 10.17 is an example of a mark made by a single stroke of a sharp pencil. Graphite particles are shown as deposited evenly and closely to conduct electricity.

The right mark in Figure 10.17 is typical of that made by several light pencil strokes. Evidently the pencil was not sharp, nor was it pressed firmly on the card. The resulting scattered deposit of graphite could not conduct enough electric current to actuate the 519 sensing circuits.

Each pencil mark should be long enough to extend all the way across the mark-sensing column. No other penciled writing can be done in the area, since these marks may be read by the machine as if they were intended for mark-sensed data. Marks may be erased if enough graphite is removed to eliminate conductivity.

Mark sensing has the advantage that data can be recorded at the site where the transaction occurs. Then, data marked at a remote location can be converted to punched holes in a data processing department without the intervening operation of keypunching.

As previously stated, up to 27 columns of information can be marked on one side of a card. However, if the cards are to be read in the read unit of the 519, column 27 is not available. Each of the

27 columns of mark sensing is read by a brush. One brush is made up of three separate brushes. The two outer ones are connected, and the center brush is wired to an exit hub on the control panel. A mark on the card conducts a small amount of electricity from the outer brushes to the center brush.

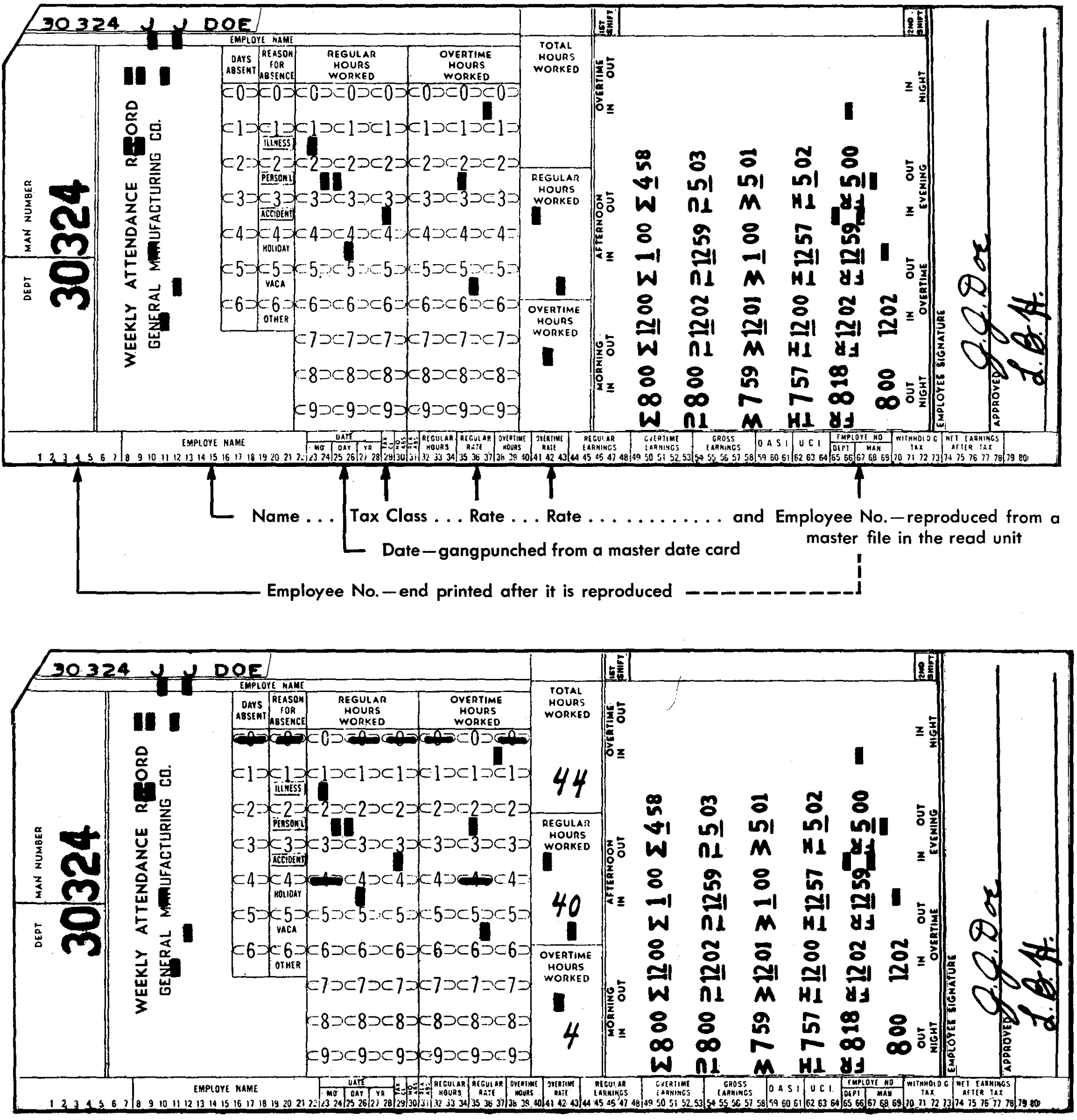
Because the pencil marks do not conduct enough electricity to operate the punch unit, the electrical impulse must be amplified. Each mark sensing entry hub on the control panel is an entry to an amplifier. The mark sensing brushes must be wired to these entries. When a brush senses a pencil mark, the small amount of electricity is amplified and emitted from the corresponding exit hubs as an impulse suitable to cause punching.

Mark-sensed punching is checked by the double-punch and blank-column detection device, installed whenever the 519 is equipped with the mark sensing feature. The device is used to detect unpunched card columns, columns containing multiple punches, or both. Consequently, if any column is not marked at its source location or if the mark fails to convert to a punched hole, an error is indicated. Similarly, if two marks are made in the same column, the machine can also be controlled to treat this situation as an error.

Usually, marked-sensed fields are marked with numerical information no more than 10 digits in length. Alphabetic letters can be marked, but in this case blank column detection only can be used in those columns so marked. Alphabetic marking is difficult to do efficiently because it is necessary to memorize the 80-column card code for each letter.

In a typical payroll application of mark sensing, attendance cards are prepared for each employee by reproducing and gangpunching from master cards. Figure 10.18 shows a sample form used for this purpose. Employee name, number, tax class, regular rate, and overtime rate are reproduced from the master deck. The date of the pay period is gangpunched into the attendance cards. During the same operation, the employee number is end-printed. After preparation, the cards are placed in a card rack with the printed ends visible at the top.

When an employee arrives at work, he removes his numbered card from the rack and inserts it in an attendance recorder. The time of day is automatically stamped on the face of the card in the space provided. The employee then returns his card to the rack where it remains during the work period. Time in and out for



Name . . . Tax Class . . . Rate . . . Rate . . . . and Employee No.—reproduced from a master file in the read unit

Date—gangpunched from a master date card

Employee No.—end printed after it is reproduced

Figure 10.18 Employee attendance record.

lunch may also be recorded. At the end of the period, the card is stamped with the time the employee leaves work.

At the end of the week, the total time worked is marked on the card with an electrographic pencil as shown in the illustration. Space is provided in another column to mark any days absent. Space is also provided to mark a *reason* code, a good example of how statistical data can be marked when required.

All employee cards are then returned to the data processing department. The marks are converted to punched holes by the 519 punch unit. Subsequently, the cards can be used in other operations to calculate and prepare the payroll.

Questions and Exercises

1. What are gangpunching, reproducing, and summary punching?
2. What is the difference between interspersed gangpunching and regular gangpunching?
3. Why must either master or detail cards in an interspersed gangpunching operation be marked with special X punching?
4. What is the function of a selector? How is it actuated?
5. In exercise 4, chapter 9, it was assumed that master pricing and requisition cards are matched and merged in preparation for pricing. What field can now be interspersed gangpunched from master to details?
6. What fields between master and details should be compared? Which comparison is really a machine check on the collating operation?
7. After gangpunching and comparing, what is the next logical step in handling the two files of records? Include these additional steps in your procedure write-up.
8. As an alternative to keypunching requisitions after they have been filled by the stock room clerk, it might be possible for the secretaries to mark sense the item number and quantity requested on the card beforehand. Design a requisition card for mark sensing with a two-digit department number, a three-digit item number, and a three-digit quantity field.
9. Devise a way to use one additional mark-sensing column for unit of measure. (Hint: Code 1 might represent "each," code 2, dozen; code 3, box; etc.) Include this field in your card design.
10. In class discussion, compare the two methods of punching the stock room requisitions: keypunching and mark sensing. Discuss the advantages and disadvantages of each method.

MACHINE CALCULATING OPERATIONS 11

Nearly every complete data processing procedure includes calculation. In commercial accounting applications, calculation is usually a matter of simple arithmetic to determine total amount derived from quantity and unit value. This category of calculation includes such common applications as billing, accounts payable, accounts receivable, inventory and production control, ledger accounting, and so on. Payroll and cost accounting require the multiplication of hours worked times rate of pay, tax rates times gross earnings, unit cost times quantity produced, and the like.

Practically all commercial calculation can be completed with the use of the four fundamental operations of addition, subtraction, multiplication, and division. In the majority of procedures, division is not usually required unless it is necessary to obtain averages of such factors as rates, cost, and sales activity. Specialized equipment is available to read factors from punched cards, calculate, and punch results in either the same card or in other designated cards, as required. Operation of the IBM 604 Electronic Calculating Punch is described in this chapter.

The 604 has two basic units: a calculator and a card reading and punching device, as shown in Figure 11.1. The units are connected by cable, so that information read from cards is transmitted

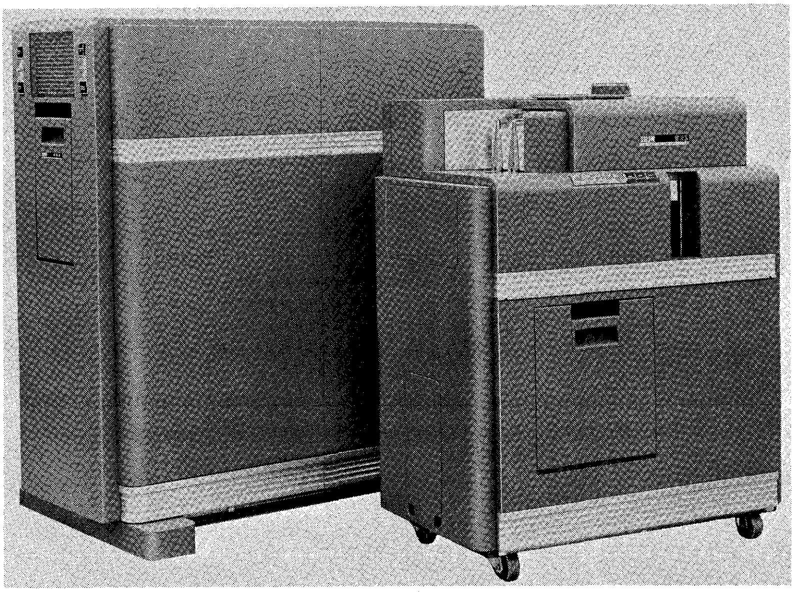


Figure 11.1 IBM 604 Electronic Calculating Punch.

to the electronic calculator where results are produced and then transmitted back to the punch for recording. Each unit is equipped with a removable control panel which can be wired to cause the machine to perform a variety of calculations. All operations are carried out with 80-column cards.

The 604 can perform a series of operations in one pass of the cards; that is, two factors can be added, the sum multiplied by a third factor, the product divided by a fourth factor, and so on. The order of calculation is predetermined by panel wiring a sequence of *program steps.* Throughput is at a constant rate of 50, 100, or 200 cards per minute, depending upon the model. The calculation capacity is as follows:

Addition and Subtraction. Totals or factors of as many as 13 positions can be accumulated in a counter.

Multiplication. An eight-digit multiplicand can be multiplied by a five-digit multiplier to produce a thirteen-digit product. Larger products can be produced by one or more additional multiplication recalculations.

Division. A thirteen-digit dividend can be divided by an eight-digit divisor to develop a five-digit quotient.

Group multiplication can be performed with either the multiplier or multiplicand as the group factor. This means that one group factor can be punched in a master card that will then serve as a constant multiplier or multiplicand for all succeeding details grouped with that master. Other group operations can also be performed, such as the accumulation of factors from a group of cards and the punching of summarized or calculated results into the last card of the group.

Figure 11.2 shows schematically the three types of processing that can be performed.

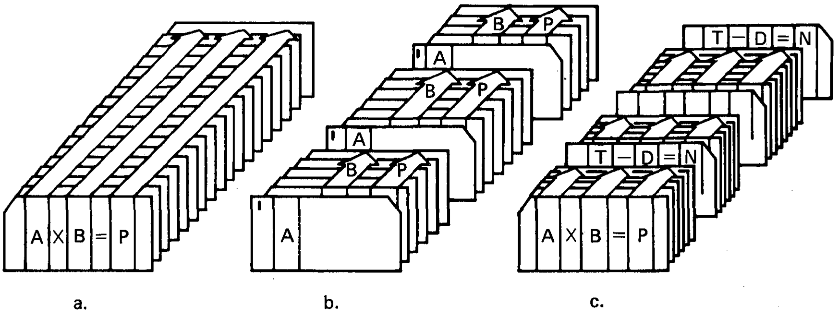


Figure 11.2 Three types of punched-card calculation.

a. Each card record is calculated as it passes through the machine. At least two factors are involved with results punched in the same card.

b. A group multiplier or multiplicand is punched into a master card as one factor. The second factor is variable and is punched in each detail card associated with the master. Details are assumed to be properly filed behind their corresponding master in a previous collating operation.

c. Calculations are made for the sum of a group of cards as well as for individual cards. For example, in a billing job, the unit price of each item is multiplied by the quantity sold to punch the billing amount in each card. As the extensions are made they are added progressively to obtain the total invoice amount.

An X-punched total card is the last card of each group with a discount rate by which the entire invoice amount is multiplied to obtain discount amount.

Discount amount is subtracted from invoice amount to obtain net amount. All three results, invoice, discount, and net are punched in the X-punched total card.

Where storage capacity permits, calculations and punching can be checked in one run. The punched result of the first calculation and the initial input factors are read back into the machine so that they may be recalculated and subtracted from the punched result. The input factors are reversed to check the machine operation. The net results of both calculations must be zero, thereby proving the accuracy of the punched results.

If storage capacity is completely used up for calculation or accumulation of totals, a second checking run can be made. Results from the second run are over-punched into the same columns as in the first run. When results are identical, as they should be, no additional holes are punched. When results differ in any card, double punching will occur in the result field. If the machine is equipped with a feature to detect double punching, the error condition can be signaled to the operator. Blank columns, or failure to punch in any column of the result field, can also be detected. By control panel wiring, either checking method can be used to stop the machine.

IBM 521 Card Read Punch

The IBM 521 Card Read Punch has three stations as shown in Figure 11.3. These are

1. A first reading station from where factors to be used in a calculation are transmitted to the electronic calculator unit.

2. A punching station where results are punched. The station can also be used for gangpunching either with or without calculation.

3. A second reading station where cards may be read for gangpunching, recalculation, or a double-punch and blank-column check.

The feed hopper has a capacity of approximately 800 cards. Input records are fed one at a time from the bottom of the hopper, 12-edge toward the machine. Cards feed continuously from one station to the next at the rate of 100 per minute (Model 1) or 50 per minute (Model 2). As shown in Figure 11.3, while one card is being read for calculation another is being punched and a third is

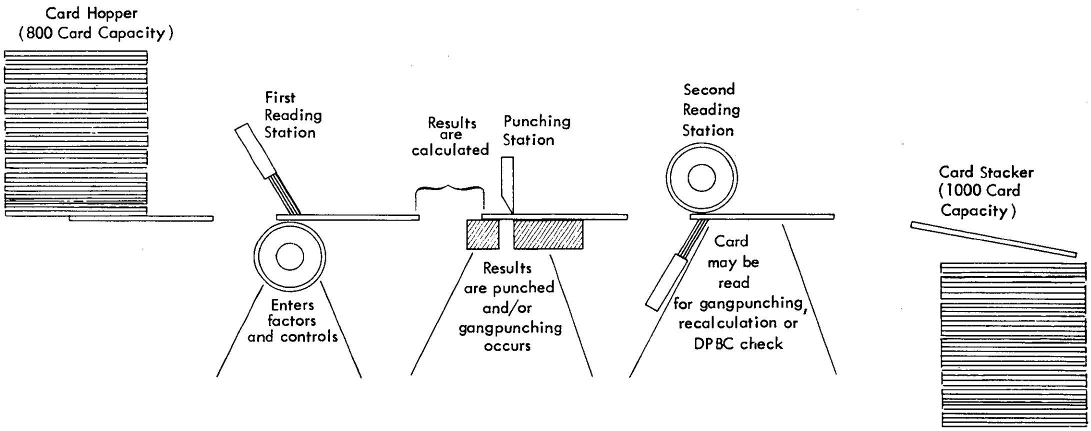


Figure 11.3 Card stations. IBM 521 Card Read Punch.

read for checking. Note that results are produced in a small fraction of a second in the interval (115 milliseconds) between the first reading station and the punching station. Punched and checked cards are automatically stacked to the right of the machine. As with nearly all types of punched-card equipment, the machine stops when the 1,000-card capacity of the stacker has been reached.

A number of keys and lights provide for operator control; these include start, stop, and reset keys. There is a red indicator light which turns on whenever the machine stops while power is on, and turns off as cards are passing through the machine.

Indicator lights also show the status of checking features. The *unfinished program light* turns on, the machine stops, and all punching is automatically suppressed in those instances when a calculation cannot be completed in 115 milliseconds. The *double-punch* and *blank-column* light is turned on by the detection of errors in a designated card field. The *zero check light* turns on and the machine stops if, in the checking operation, the punched result subtracted from the recalculated result does not equal zero. The *product overflow light* turns on and the machine stops if the result of a calculation exceeds the number of card columns provided for punching. All check lights operate only when conditioned by corresponding proper control panel wiring.

IBM 604 Electronic Calculating Unit

The 604 performs all arithmetic calculations. There are five basic parts to the unit:

1. Electronic storage units
   Factor storage
   Multiplier-quotient storage
   General storage
2. Common channel
3. Column shift unit
4. Counter
5. Program unit

The *factor storage* unit (FS) contains 16 positions for the storage of input factors read from punched cards. Storage is divided into two 3-position and two 5-position units. Storage may also be used to retain intermediate results during calculation.

The *multiplier-quotient* storage unit (MQ) can hold up to a 5-position multiplier factor received either from a card, from another storage unit, or from the counter. During division it is designed to receive up to a 5-position quotient as it is developed. It may also be used for the temporary retention of other factors, either read from a card or developed during the calculating process.

The *general storage* unit (GS) is intended primarily for the storage of calculated results until punch time. It may also be used for input factors from the card. The 16 positions of general storage are divided into two 3-position and two 5-position units.

All of the electronic storage units may be considered as working storage during a series of calculating steps. Once a factor in one of the storage units has served its purpose, the unit may be reused for storage of some intermediate result. However, only the counter and the general storage are output for punching results.

An 8-position *common channel* serves as a factor path between all the storage units and the counter. Factors may be moved in groups of three, five, six, or eight positions at a time under control of assignment wiring on the control panel. Thus, only one factor of up to eight positions may be transferred over the common channel during any one program step. Detailed description of channel operation will not be included here.

The *column shift* unit permits factors that are transferred from a storage unit to the counter or from one storage unit to another to be shifted as many as five positions to the left of their normal entry position. As many as five positions can be dropped when results are transferred from the counter to a storage unit. An entry shift (read-units-into) moves all positions of the factor as many as five positions to the left and prevents positions from the right from reading in. An exit shift (read-units-out-of) permits the dropping of as many as five places from the right of a factor. Shifting is controlled by panel wiring.

Shifting which is inherent in multiplication and division is accomplished automatically by the column shift unit on multiply and divide steps.

A 13-position *counter* is used to perform addition, subtraction, multiplication, and division. Factors cannot be read directly from a card into the counter, but results developed in the counter can be punched into a card.

The *program unit* supplies electronic impulses to control the arithmetic functions and provide read-in and read-out signals to the storage units and counter. After data from the card is read into storage units, calculation is started automatically by an impulse originating from the exit hubs of program step 1. The machine then proceeds sequentially through the remaining steps. Twenty program steps are available with the standard 604. This number may be increased to 40 or 60 as optional features.

The 604 is provided with switches and keys to control machine operation and with lights to indicate the status of conditions that affect operations. Of these, only the program test key and the program test light will be discussed here.

Depressing the *program test key* makes it possible to operate the 604 manually one program step at a time. The *program test light* turns on to indicate that the machine is in test rather than run status. By using a test card and a planning chart (explained in the next section), the operator can take a calculation through the machine in complete detail. The contents of the storage units are visible on a display panel located on the calculating unit.

604 Display Panel

The 604 is equipped with an operation display panel—an arrangement of small neon bulbs. When lighted, selected bulbs show all factors stored in the machine and the operation being performed during any given program step. The panel is used as an aid in locating errors in procedure planning or control panel wiring.

To check wiring, a test card is fed through the card read punch. The punched results are then checked against the calculation planning chart. If there is disagreement, it is usually the best practice to check the wiring first. If no errors are discovered, each program step in the calculation can be traced as follows:

1. The machine is placed in test status by depressing the program test key.

2. The test card is placed in the hopper of the card read punch and fed two cycles. This reads all input factors into their assigned storage units.

3. The calculation is performed one step at a time by depressing the program advance key. The machine results appear on the display panel where they can be visually compared with the manual result on the planning chart. This procedure is repeated for all steps in the calculation until the error is located.

4. The machine can be returned to normal operation by depressing the program test key a second time.

Each position of factor storage, general storage, multiplier-quotient storage, and the counter is represented by four neons arranged in a vertical column. The top neon is given a value of 1, the second a value of 2, the third a value of 4, and the bottom neon a value of 8. The digit stored in any one position is mentally calculated by adding the values of the lighted bulbs in that column. The arrangement to display the four digits 9637 is shown in Figure 11.4. The machine actually uses binary coded decimal representation for numerical values. This code is explained more fully in a later chapter dealing with the IBM System/3 computer.

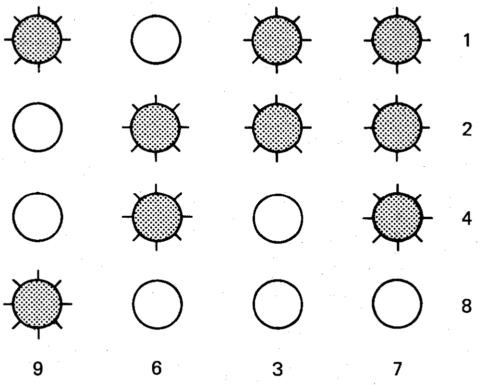


Figure 11.4 Schematic. 604 display panel.

Other lights on the panel indicate the number of the program step being executed, the operation being performed, and the position of shift when a factor is transferred from one storage unit to another.

Digits in the counter are represented as 9s complements of a positive number; a negative number appears as a true figure with a minus neon turned on. Both 9s complements and true figures are represented as binary coded decimals on the display panel.

604 Operation Planning

All calculation performed by the 604 must be accomplished within the predetermined number of program steps available in the machine. In any given step, the calculating unit can perform one of a specified list of operations:

Counter Read-in Plus. The contents of a storage unit are added to the contents of the counter. The particular storage unit is specified by control panel wiring.

Counter Read-in Minus. The contents of the specified storage unit are subtracted from the contents of the counter.

Counter Readout. The contents of the counter are transferred to a storage unit. The counter is not reset or cleared. Storage unit contents are cleared by the read-in.

Counter Readout and Reset. The counter reads out to a specified storage unit and is cleared at the beginning of the next program step.

Balance Test for Step Suppression. The plus or minus status of the counter can be used to activate a corresponding set of suppress on plus or minus balance hubs on the control panel. These hubs become active on the next program step and continue to be active until impulsed again or until the end of calculate time.

Suppress on Plus Balance. When the above balance test is made, the counter is either plus or minus. If it is plus, suppress on plus balance hubs may be used to suppress succeeding program steps for that card.

Suppress on Minus Balance. If the above balance test shows the counter to be minus, suppress on minus balance hubs may be used to suppress succeeding program steps for that card.

The three program suppression operations just described provide the machine with the ability to condition its operation depending upon the positive or negative balance of results. Thus, the machine can follow alternate paths through a procedure as directed by the actual results encountered in a series of calculations.

Multiply Plus. The contents of a factor storage storage are multiplied by the contents of the multiplier-quotient storage. The product is developed in the counter and is plus. Multiplication in the 604 is accomplished by repetitive addition. The multiplicand in factor storage is automatically added and offset internally in the counter the number of times called for by the multiplier. The maximum number of positions in the product is equal to the sum of the positions in the multiplier (MQ) and the multiplicand (FS).

Multiply Minus. The product developed in the counter will be minus. All other conditions of the previous operation, multiply plus, apply.

Divide. The contents of the counter are divided by the contents of the designated factor storage. The quotient is developed in the MQ unit. The remainder is left in the counter and can be cleared by counter readout and reset.

Division on the 604 is the process of reducing the dividend by subtracting the divisor the number of times that it takes to reduce the dividend to a remainder that is less than the divisor. The operation is automatic once the divisor has been impulsed to readout and the divide hubs have been impulsed.

Half Adjust. A digit 5 is added to the position of the counter designated by panel wiring. The operation forces a carry of 1 to the next most significant position in the counter when the sum of the 5 and the digit to which it is added is greater than nine. The effect is one of rounding a result to the nearest penny, nickel, dime, or dollar.

Zero Check. Corresponding hubs on the control panel can be used to check the contents of the counter for a zero balance.

Figure 11.5 is the planning chart used to determine the sequence of operations required for a given job. The chart is divided into (1) a vertical arrangement of the entire calculate cycle of program steps and (2) a horizontal arrangement of the various storage units into which the factors may be entered.

Program Number. The first column on the chart identifies the factors to be read in from cards (top line) and provides space for writing in the program step number. Results to be punched into cards are identified in the bottom line.

Operation Notes. This column can be used for notes of explanation about each operation.

Read-Units-Into, Out-Of. Read-units-into shows the counter or storage position *into* which the units position of the factor is read. Read-units-out-of shows the counter or storage position *from* which the units position of the factor is read. The column is normally left blank unless the shift unit is impulsed.

Program Suppress. Program steps may be suppressed depending upon conditions encountered in the sequence of operations; for example, a positive or a negative balance.

Program Number. This column is the same as the first column.

Factor Storage. The four factor storage units are spaced in this column. Two units have three positions each; two have five positions. Transfer of data into or out of these units is indicated on the line corresponding to the program step.

Note that in the heading of the factor storage column, an *assignment* indication is made. For example, factor storage 1 may be assigned to a 6-4 or an 8-6 position. Depending upon which assignment is wired on the control panel, this means that the digits stored in this unit will at all times be transferred as though they were the leftmost digits of either a six-position or eight-position number. This feature has the effect of coupling the storage units together for the purpose of transferring data during the calculation sequence. Factor storage 3 may also be given an 8-6 assignment.

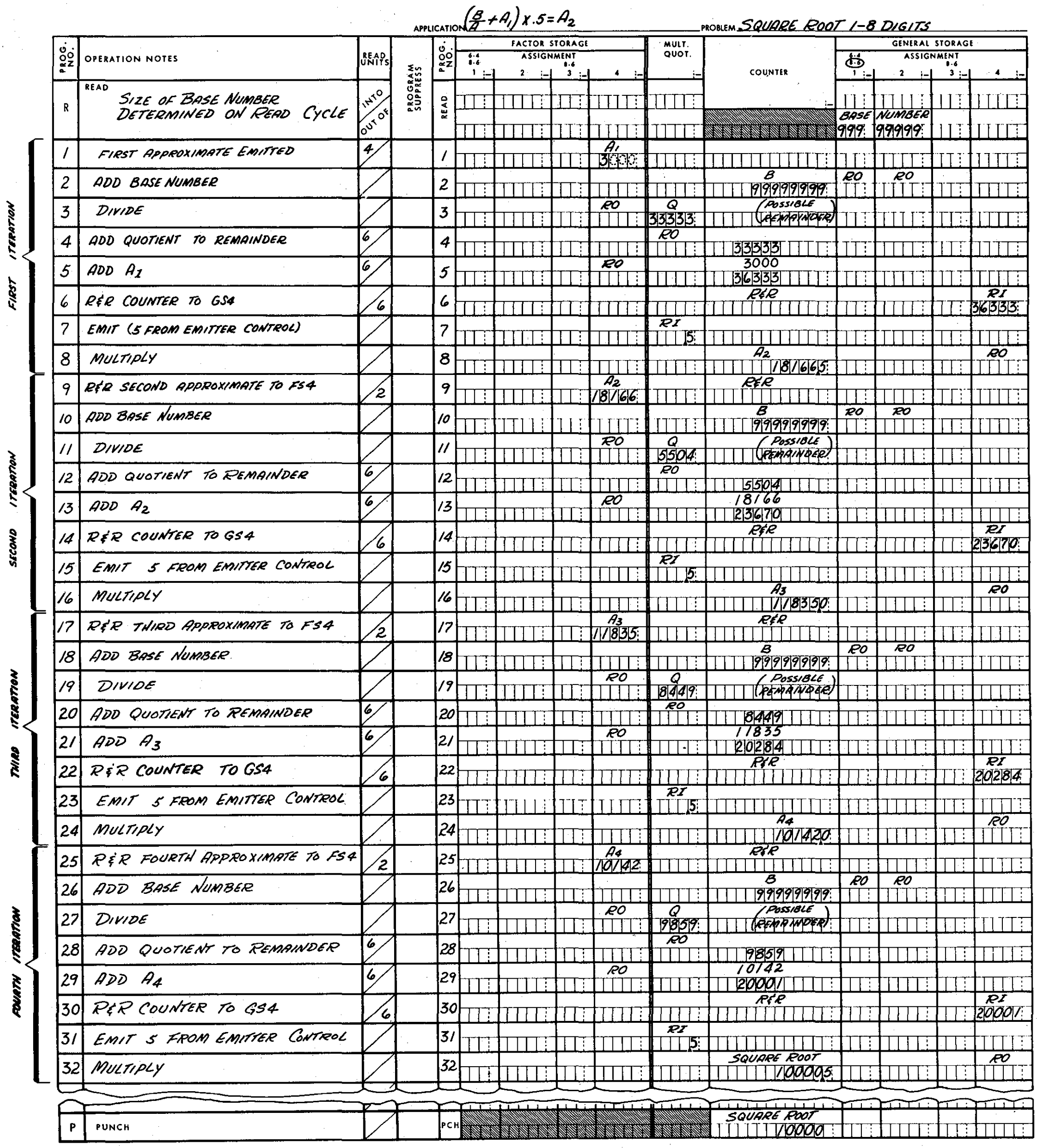
Assume that factor storage 1 contains the digits 123 and is assigned the 8-6 position. Also assume that factor storage 2 contains the digits 45678. If FS 1 and FS 2 are both impulsed to read out into the counter in the same program step, the counter will receive the number 12345678.

Further assume that FS 1 again contains the digits 123 but is given the assignment of 6-4. FS 3 contains the digits 456 and is not assigned. If FS 1 and FS 3 are both impulsed to read out in the same program step, the data transferred will be the number 123456.

On the planning chart, the factor storage units are shaded on the bottom punch line to indicate that they cannot be used for output data to be punched into a card.

Counter. The 13 positions available in the counter are shown in this column. The spaces may be used to write the results of a calculation as determined manually. The top or read line of the column is shaded to indicate that data cannot be entered directly into the counter from a card.

General Storage. The four general storage units are spaced in this area. Two units have three positions of storage; two have five



APPLICATION $(\frac{B}{A} + A_1) \times .5 = A_2$ PROBLEM SQUARE ROOT 1-8 DIGITS

	PROG. NO.	OPERATION NOTES	READ UNITS INTO	READ UNITS OUT OF	PROGRAM SUPPRESS	PROG. NO.	FACTOR STORAGE ASSIGNMENT 1	FACTOR STORAGE 2	FACTOR STORAGE 3	FACTOR STORAGE 4	MULT. QUOT.	COUNTER	GENERAL STORAGE ASSIGNMENT 1	GENERAL STORAGE 2	GENERAL STORAGE 3	GENERAL STORAGE 4
	R	READ SIZE OF BASE NUMBER DETERMINED ON READ CYCLE				READ							BASE 999	NUMBER 99999		
FIRST ITERATION	1	FIRST APPROXIMATE EMITTED	4			1				A₁ 3000						
	2	ADD BASE NUMBER				2						B 99999999	RO	RO		
	3	DIVIDE				3				RO	Q 33333	(POSSIBLE REMAINDER)				
	4	ADD QUOTIENT TO REMAINDER	6			4					RO	33333				
	5	ADD A_1	6			5				RO		3000 36333				
	6	R&R COUNTER TO GS4		6		6						R&R				RI 36333
	7	EMIT (5 FROM EMITTER CONTROL)				7					RI 5					
	8	MULTIPLY				8						A_2 181665				RO
SECOND ITERATION	9	R&R SECOND APPROXIMATE TO FS4		2		9				A_2 18166		R&R				
	10	ADD BASE NUMBER				10						B 99999999	RO	RO		
	11	DIVIDE				11				RO	Q 5504	(POSSIBLE REMAINDER)				
	12	ADD QUOTIENT TO REMAINDER	6			12					RO	5504				
	13	ADD A_2	6			13				RO		18166 23670				
	14	R&R COUNTER TO GS4		6		14						R&R				RI 23670
	15	EMIT 5 FROM EMITTER CONTROL				15					RI 5					
	16	MULTIPLY				16						A_3 118350				RO
THIRD ITERATION	17	R&R THIRD APPROXIMATE TO FS4		2		17				A_3 11835		R&R				
	18	ADD BASE NUMBER				18						B 99999999	RO	RO		
	19	DIVIDE				19				RO	Q 8449	(POSSIBLE REMAINDER)				
	20	ADD QUOTIENT TO REMAINDER	6			20					RO	8449				
	21	ADD A_3	6			21				RO		11835 20284				
	22	R&R COUNTER TO GS4		6		22						R&R				RI 20284
	23	EMIT 5 FROM EMITTER CONTROL				23					RI 5					
	24	MULTIPLY				24						A_4 101420				RO
FOURTH ITERATION	25	R&R FOURTH APPROXIMATE TO FS4		2		25				A_4 10142		R&R				
	26	ADD BASE NUMBER				26						B 99999999	RO	RO		
	27	DIVIDE				27				RO	Q 9859	(POSSIBLE REMAINDER)				
	28	ADD QUOTIENT TO REMAINDER	6			28					RO	9859				
	29	ADD A_4	6			29				RO		10142 20001				
	30	R&R COUNTER TO GS4		6		30						R&R				RI 20001
	31	EMIT 5 FROM EMITTER CONTROL				31					RI 5					
	32	MULTIPLY				32						SQUARE ROOT 100005				RO
	P	PUNCH				PCH						SQUARE ROOT 10000				

Figure 11.5 Calculating planning chart.

positions. The units may be assigned in the same manner as the factor storage units.

The horizontal space given to a program step for the storage units and counter is divided into two sections. The upper section can be used to write in symbols, letters, or words that identify a factor or result. The lower section is spaced for the actual number of positions in each unit. An extra space to the right is provided to note the sign of the factor or result.

Control Panels

The 604 Electronic Calculating Unit and the 521 Card Read Punch are equipped with separate control panels which can be manually wired to adapt the machine to specific procedures. Panels are removable so that the units can be readily changed from one job to another by inserting properly wired panels. Figure 11.6 is a schematic of the panels with all exit and entry hubs labeled. Shaded areas indicate optional features.

Card reading, selection of data for entry into storage, and card punching are controlled by the 521 or 541 panels. All operations of calculation are controlled by the 604 panel.

Basic Operation

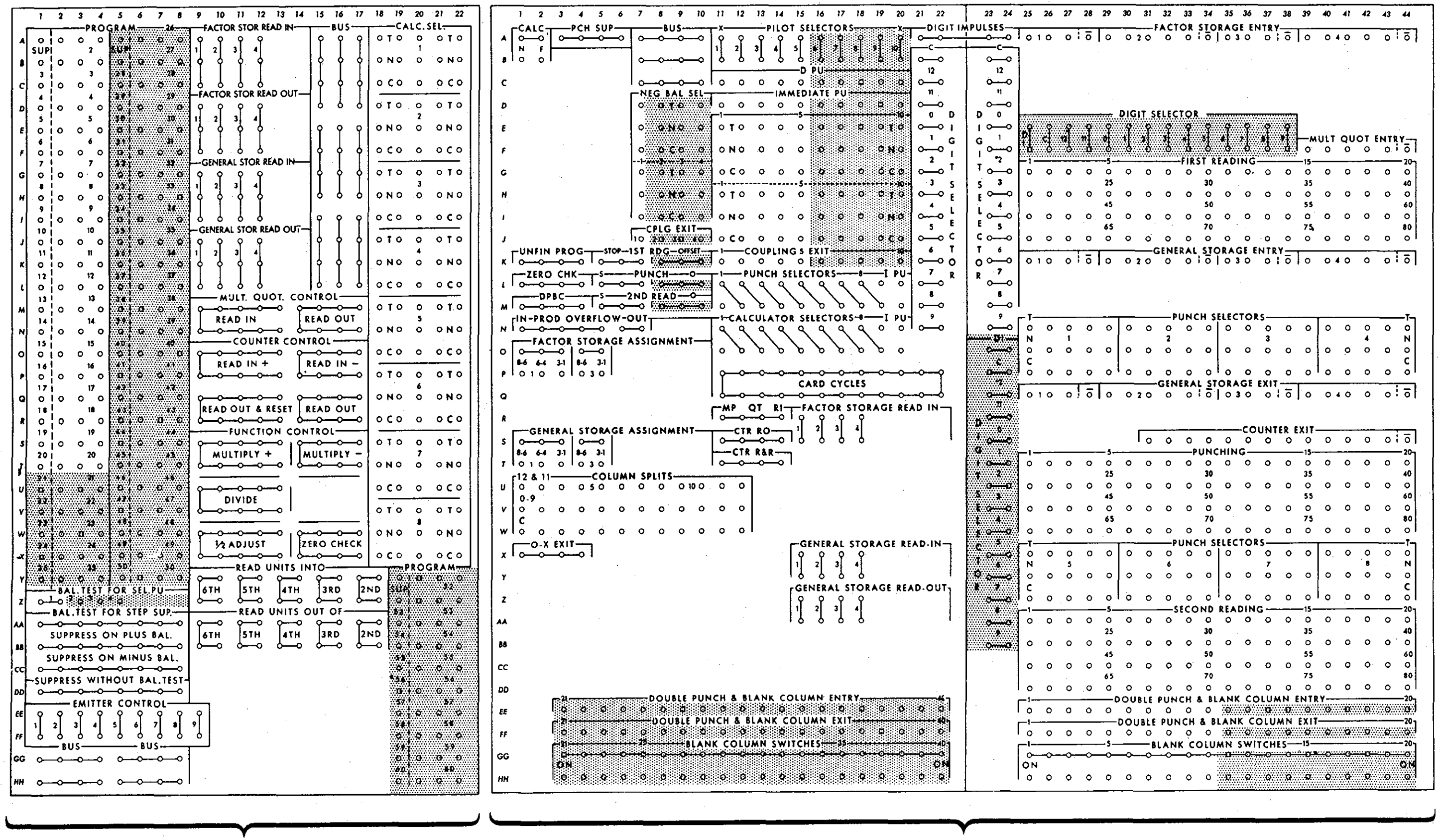
The following sections describe the basic data processing operations that can be performed by the 604. Some panel wiring diagrams are shown, but these are not discussed in any detail. Various IBM machine manuals are available for this purpose.

In studying these sections, the student's main emphasis should be placed upon the data handling capabilities of the machine, because many of these serve to illustrate methods of factor manipulation used by computer and other automatic calculating devices.

Crossfooting: $A - B = \pm T$

The 604 executes a series of 20 program steps during each card cycle. Program steps are electronic cycles that occur successively. All steps are taken whether or not they are actually used to complete a calculation. As many as 40 or 60 steps can be executed by machines equipped with these optional features.

The number of factors that may be crossfooted is limited only by the amount of data that can be stored during a read cycle. In this example, a simple crossfooting operation is performed to subtract factor *B* from factor *A*. The result *T* may be either plus or minus; if minus, the true amount will be punched with an X punch over the units position of the card field. Factor *A* is punched in columns 31 through 35 and factor *B* in columns 25 through 27. The result *T* will be punched in columns 70 through



604 Calculator Control Panel

521-541 Card Read Punch Control Panel

Figure 11.6 604 and 521 control panels.

74. The wiring is shown in Figure 11.7, the planning chart in Figure 11.8. Typical values of the factors are manually inserted to plan the calculation.

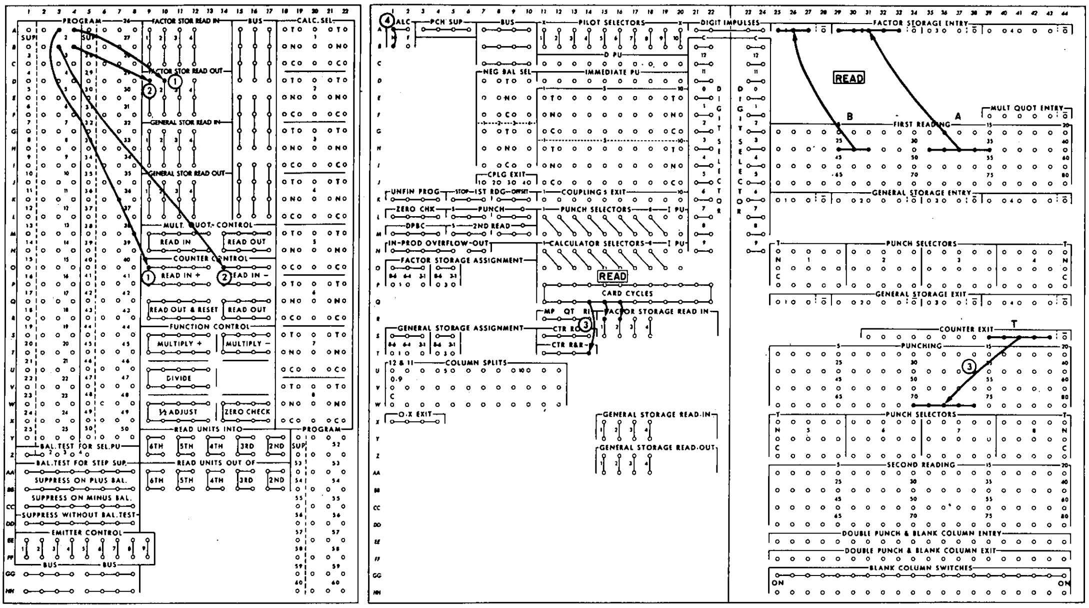
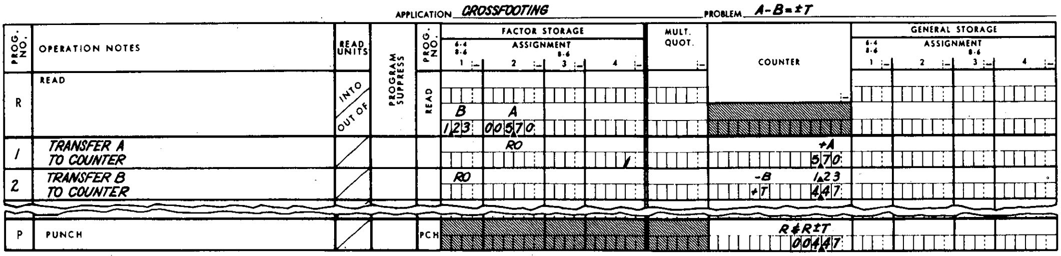


Figure 11.7 Control panel wiring. Crossfooting.



APPLICATION CROSSFOOTING PROBLEM A-B=±T

PROG. NO.	OPERATION NOTES	READ UNITS	PROGRAM SUPPRESS	PROG. NO.	FACTOR STORAGE ASSIGNMENT 1	2	3	4	MULT. QUOT.	COUNTER	GENERAL STORAGE ASSIGNMENT 1	2	3	4
R	READ	INTO OUT OF		READ	B 1.23	A 005.70								
1	TRANSFER A TO COUNTER					RO				+A 5.70				
2	TRANSFER B TO COUNTER				RO					-B 1.23 +T 4.47				
P	PUNCH			PCH						R≠R±T 004.47				

Figure 11.8 Planning chart. Crossfooting.

Read

Factor *B* is placed in factor storage 1; factor *A* in factor storage 2.

1. On the first program step, *A* is read out of FS 2 and placed in the counter (read-in plus). Note that the position of the decimal point is also shown for planning purposes. Whole numbers may be crossfooted with decimal numbers by shifting the transfer

the proper number of positions to the right or left (read-units-into, out-of).

Punch

All remaining program steps are not used. As the card passes the punching station in the card read punch, the result is punched in columns 70 through 74.

Multiplication: *A* × *B* = *P*

To do multiplication, the multiplicand is placed in any storage unit; the multiplier in the MQ unit. Factors can be read into the units directly from a card, or they may be developed as results from previous calculations. The designated storage unit is impulsed to read out and in the same program step, the machine is impulsed to multiply. The product is developed in the counter.

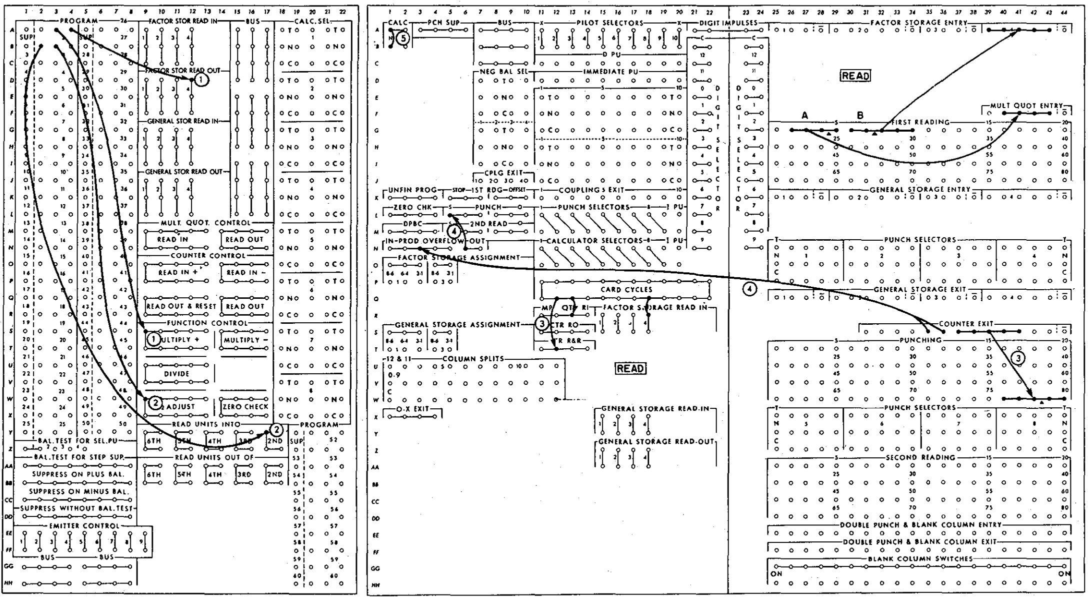


Figure 11.9 Control panel wiring. Basic multiplication.

Figure 11.9 is a wiring diagram for basic multiplication. Multiplicand *B* is assumed to be an hourly rate field read from card columns 6 through 10 directly into factor storage 4. Multiplier *A* is an hours worked field read from columns 2 through 5 into the multiplier-quotient. Product *P* (hourly earnings) is to be punched into columns 76 through 80 from the counter. Note that the last two counter positions are not punched and that any significant positions to the left of the five-digit field are wired to cause a product overflow check. If the product exceeds the five positions expected, the machine stops when the card in error reaches the stacker. The product overflow light on the Card Read Punch comes on.

Figure 11.10 is the planning chart for the example of multiplying *hours* X *rate* = *earnings*.

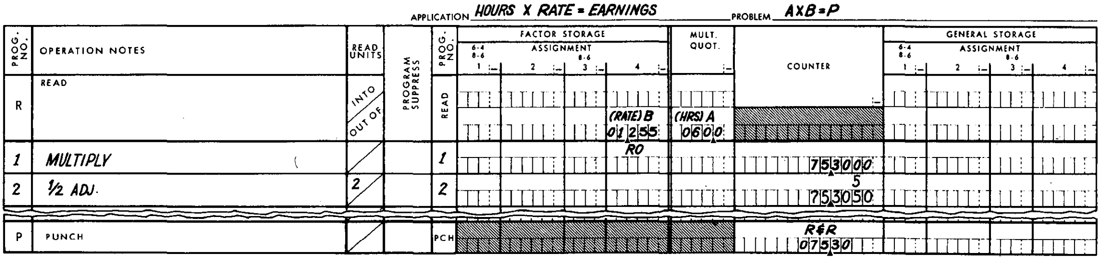

APPLICATION HOURS X RATE = EARNINGS
PROBLEM AXB=P
PROG. NO.
OPERATION NOTES
READ UNITS
PROGRAM SUPPRESS
FACTOR STORAGE ASSIGNMENT
6-4 8-6
1
2
3
4
MULT. QUOT.
COUNTER
GENERAL STORAGE ASSIGNMENT
R
READ
INTO
OUT OF
(RATE) B
01255
RO
(HRS) A
0600
1
MULTIPLY
753000
2
½ ADJ.
5
753050
P
PUNCH
PCH
R$R
07530

Figure 11.10 Planning chart. Basic multiplication.

Read

The hourly rate (multiplicand *B*) is read into FS 4. The rate has three decimal places, for example, $1.255 per hour. The hours worked (multiplier *A*) field is read into the MQ. The field has one decimal place, as shown.

1. FS 1 is read out and the machine is impulsed to multiply. The product is developed in the counter. The product has four decimal places.

2. The product is half adjusted in the second position by wiring read-units-into the second entry hubs.

Punch

Five positions of the product are punched into card columns 76 through 80.

Division: $A \div B = Q$

The 604 provides the capacity to divide a 13-digit dividend by an eight-digit divisor to produce a five-digit quotient in a single program step. If these limits are exceeded, the operation can be done in parts with several program steps.

The dividend is placed in the counter; the divisor in any storage unit. The quotient is developed in the MQ. Decimal point alignment between dividend and divisor is accomplished by shifting the dividend the required number of places as it is placed in the counter. The divisor may also be shifted as required.

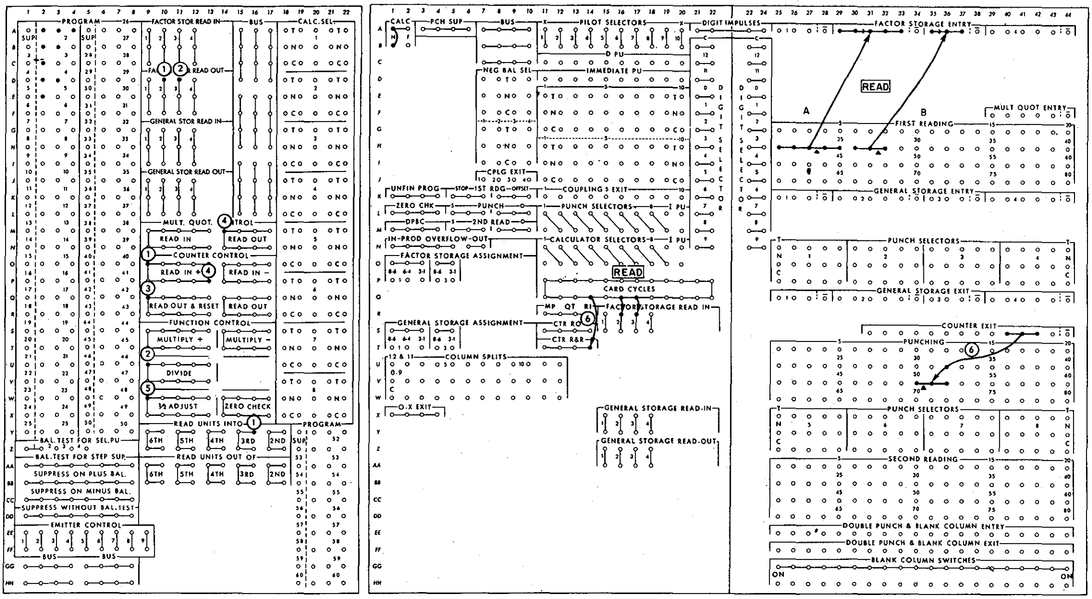
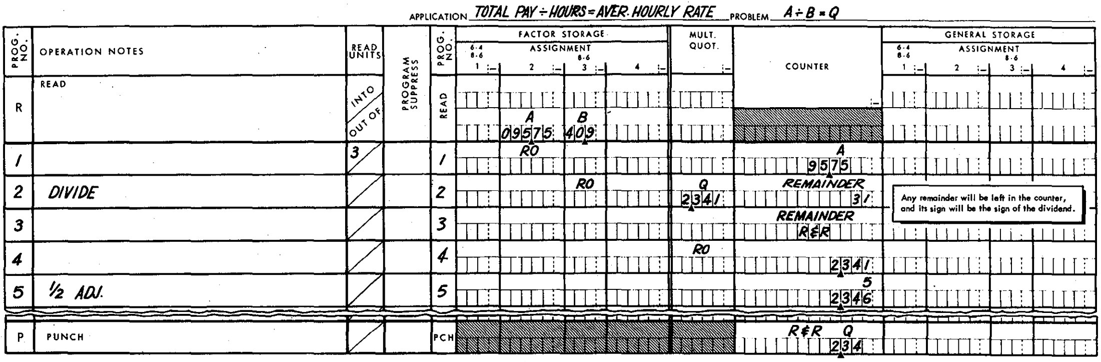


Figure 11.11 Control panel wiring. Basic division.

Figure 11.11 is the wiring diagram for an example of basic division. The example illustrates the division of total pay (*A*) divided by hours (*B*) to produce an average hourly rate. *A* is punched in columns 21 through 25 of the card; *B* is punched in columns 26



APPLICATION TOTAL PAY ÷ HOURS = AVER. HOURLY RATE PROBLEM A ÷ B = Q

PROG. NO.	OPERATION NOTES	READ UNITS	PROGRAM SUPPRESS	PROG. NO.	FACTOR STORAGE ASSIGNMENT 1	2	3	4	MULT. QUOT.	COUNTER	GENERAL STORAGE ASSIGNMENT 1	2	3	4
R	READ	INTO / OUT OF		READ		A 09575	B 409			▨				
1		3		1		RO				A 9575				
2	DIVIDE			2			RO		Q 2341	REMAINDER 31	Any remainder will be left in the counter, and its sign will be the sign of the dividend.			
3				3						REMAINDER R&R				
4				4					RO	2341				
5	½ ADJ.			5						5 2346				
P	PUNCH			PCH	▨	▨	▨	▨	▨	R&R Q 234				

Figure 11.12 Planning chart. Basic division.

through 28. The quotient Q is to be punched as output from the calculation in columns 50 through 52. Figure 11.12 is the planning chart.

Read

The dividend A is read into FS 2 and the divisor B into FS 3.

1. On the first program step, the dividend A in FS 2 is read out into the counter (read-in plus). The contents of FS 2 are shifted to the third position of the counter to align the decimal points to produce a three-decimal quotient.

2. The divisor B is read out of FS 3 and the machine is impulsed to divide. The quotient is automatically developed in the MQ.

3. The dividend remainder is cleared out of the counter by impulsing the counter to read out and reset. The remainder can be saved, if needed, by impulsing a storage unit to read-in during the same program step. Note that the counter must be cleared in a separate step; it does not clear on a read-in as the counters do.

4. The quotient is transferred to the counter in preparation for half adjustment to the nearest cent.

5. The quotient is rounded to the nearest cent by adding 5 to the units position of the counter. The remaining program steps are not used.

Punch

The average hourly rate is punched in columns 50 through 52.

Successive Calculations: $(A \times B = P) \div C = R$

Successive calculations can be made involving any of the arithmetic and data handling operations the machine can perform. The number of calculations is limited by storage capacity for input factors, availability of storage for intermediate results, and storage for accumulating final results to be punched in cards.

Figure 11.13 is the wiring diagram for the calculation of the formula $(A \times B = P) \div C = R$ or the example of (*Pieces* × *unit standard time* = *total standard time*) ÷ *actual time* = *% over or under standard.*

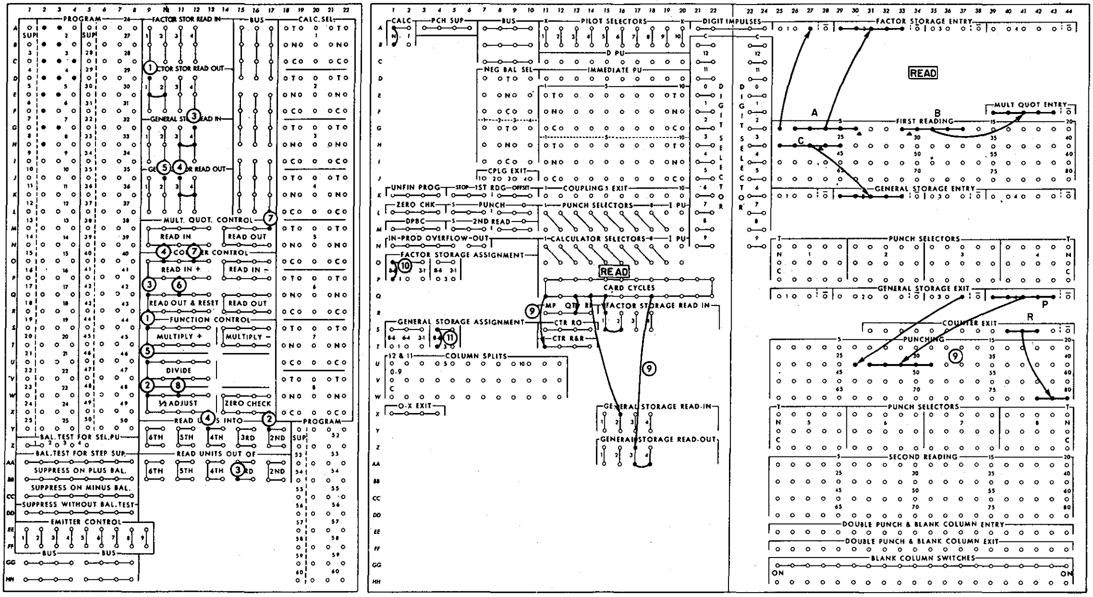
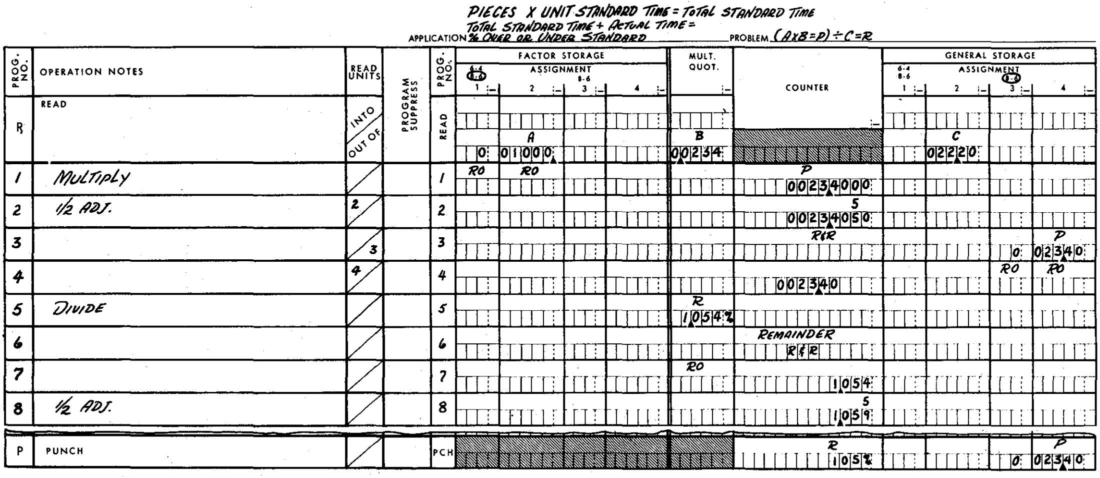


Figure 11.13 Control panel wiring. Successive calculation.

A card field *A* (columns 1 through 6) represents the number of pieces of an item that a worker has produced in an actual period of time. Field *C* in columns 21 through 25 is the actual time. *A* is a whole number; *C* has two decimal places for recording to hundredths of an hour.

The purpose of the formula calculation is to compare actual production against an established standard rate, represented by



PIECES X UNIT STANDARD TIME = TOTAL STANDARD TIME
TOTAL STANDARD TIME ÷ ACTUAL TIME =
APPLICATION % OVER OR UNDER STANDARD PROBLEM (A×B=P)÷C=R

PROG. NO.	OPERATION NOTES	READ UNITS	PROGRAM SUPPRESS	PROG. NO.	FACTOR STORAGE ASSIGNMENT 6-4 8-6 1	2	3	4	MULT. QUOT.	COUNTER	GENERAL STORAGE ASSIGNMENT 6-4 8-6 1	2	3	4
R	READ	INTO OUT OF		READ	0	A 01000			B 00234	▨		C 02220		
1	MULTIPLY			1	RO	RO				P 00234000				
2	1/2 ADJ.	2		2						5 00234050				
3		3		3						R&R			0	P 02340
4		4		4						002340			RO	RO
5	DIVIDE			5					R 10542					
6				6						REMAINDER R&R				
7				7					RO	1054				
8	1/2 ADJ.			8						5 1059				
P	PUNCH			PCH	▨	▨	▨	▨	▨	R 1059			0	P 02340

Figure 11.14 Planning chart. Successive calculation.

field *B* in columns 9 through 13. The result *R* shows the percentage over or under standard. In some payroll plans, a bonus is paid to the worker when standards are met or exceeded.

Figure 11.14 is the planning chart.

Read

Field *A* is read into FS 1 and FS 2, *B* into the MQ unit, and *C* into GS 2.

Note that FS 1 is assigned to positions 8-6, effectively coupling FS 1 and FS 2 into an eight-position storage unit. This is necessary because the field *A* is a six-position field and cannot be contained in FS 2 alone. GS 3 is also given an 8-6 assignment to contain an intermediate result which will exceed the capacity of the five-position GS 4. Decimals are placed in the factors as shown below.

1. Factor *A* is multiplied by factor *B* by reading out FS 1 and FS 2 and by impulsing the multiply plus. Since there are four decimals in *B* and none in *A*, there will be four decimal places in the product, total standard time.

2. Total standard time (*P*) is adjusted to the nearest hundredth of an hour by half adjusting. A digit 5 is added to the second position of the counter in this program step.

3. The product *P* must be stored until punching time. Therefore, the counter is read out and reset with two positions dropped, into GS 3 and GS 4. The positions are dropped by impulsing the read-units-into the third position.

4. The product *P* must now be transferred back to the counter as a dividend in preparation for the next step in calculating the formula. *P* is read out of GS 3 and GS 4 into the counter. Two decimals are required in the quotient, plus two in the divisor, plus one for half adjustment, making it necessary to place the decimal in the fifth position of the counter. Since *P* has only two decimal places, it is offset four positions in the counter to align with the predetermined decimal point. This is done by impulsing read-units-into the fourth position.

5. *P* is divided by *C* by impulsing the divide hubs on the control panel and reading out *C* from GS 2 in the same program step. The quotient is developed automatically in the *MQ* unit.

6. The dividend remainder is not required and is cleared out of the counter by a readout and reset operation.

7. The quotient R is transferred to the counter for half adjusting.

8. The quotient is adjusted to the nearest percent.

Punch

The fields *P and R* are punched as output factors in columns 26 through 31 and columns 78 through 80 in the card.

Sign Control

Multiplication or division of factors identified by plus or minus signs can be performed automatically on the 604.

Signs of factors are entered, together with the factors, into any of the storage units that accept information from a card. When the signs are represented by X punching over the units position of the field, no special control panel wiring is required for sign control. When the X is punched in any other card column, the column entry to storage must be specially wired to segregate the X, which is then wired to a sign control position for each storage unit. If an X is punched over the units position of a field for reasons other than sign control, the X must be eliminated when the field is read into storage. This is done by wiring a device on the panel called a *column split.*

The rightmost position of each storage unit is reserved for sign control, as shown on the planning chart. This position accepts any digit or an X to identify the positive or negative value of a factor. By an internal column split, an X punch over the units position of a factor is automatically entered in the sign control position of the unit, with no wiring required. A digit can be used for sign control by wiring from the card column containing the digit through a special *digit selector* on the panel.

The calculation of a positive or negative result as required by the combination of signs is completely automatic, including the transfer of factors or results from one storage unit to another. There is a special channel for the transfer of signs.

When multiplying, the product is positive or negative, depending upon the combination of signs for the multiplier and multiplicand, and whether multiply plus or multiply minus is used. The following combinations are possible.

Multiply Plus	Multiply Minus
$(+A) \times (+B) = +P$	$(+A) \times (+B) = -P$
$(-A) \times (-B) = +P$	$(-A) \times (-B) = -P$
$(+A) \times (-B) = -P$	$(+A) \times (-B) = +P$
$(-A) \times (+B) = -P$	$(-A) \times (+B) = +P$

Sign control for division follows the standard rules of algebra.

Approximate Square Root

The square root of a base number containing eight digits or less may be produced on the 604 by four iterations of the following formula:

$$\left(\frac{B}{A1} + A1\right) \times 0.5 = A2$$

where B = base number for which the square root is to be found
$A1$ = first approximate square root
$A2$ = second approximate square root.

A1 is set up in the machine from a digit emitter. *A1* may be 3, 30, 300, or 3,000 depending upon the number of significant digits in the base number. If the base number has one or two digits, 3 is used. For a three- or four-digit base number, 30 is used. For a five- or six-digit base number 300 is used; for a seven- or eight-digit base number, 3,000 is used. Theoretically, any number can be used for the first approximate square root. The number 3 is chosen because greater accuracy is obtained when the root is carried beyond the decimal point.

As indicated on the planning chart in Figure 11.15, *A2* and *A3* are the successive approximates developed by the machine on each iteration. *A4* is the approximate square root.

Some base numbers do not require four iterations, but a maximum of four is required for any base number from one to eight digits. For example, the square root of 16 is developed in one iteration:

$$\frac{16}{3} + 3 \times 0.5 = 4$$

APPLICATION $\left(\frac{B}{A}+A_1\right)x.5=A_2$ PROBLEM SQUARE ROOT 1-8 DIGITS

	PROG. NO.	OPERATION NOTES	READ UNITS	PROGRAM SUPPRESS	PROG. NO.	FACTOR STORAGE ASSIGNMENT 1	2	3	4	MULT. QUOT.	COUNTER	GENERAL STORAGE ASSIGNMENT 1	2	3	4
	R	READ SIZE OF BASE NUMBER DETERMINED ON READ CYCLE	INTO / OUT OF		READ							BASE 999	NUMBER 99999		
FIRST ITERATION	1	FIRST APPROXIMATE EMITTED	4/		1				A_1 3000						
	2	ADD BASE NUMBER	/		2						B 99999999	RO	RO		
	3	DIVIDE	/		3				RO	Q 33333	(POSSIBLE REMAINDER)				
	4	ADD QUOTIENT TO REMAINDER	6/		4					RO	33333				
	5	ADD A_1	6/		5				RO		3000 36333				
	6	R&R COUNTER TO GS4	/6		6						R&R				RI 36333
	7	EMIT (5 FROM EMITTER CONTROL)	/		7					RI 5					
	8	MULTIPLY	/		8						A_2 181665				RO
SECOND ITERATION	9	R&R SECOND APPROXIMATE TO FS4	/2		9				A_2 18166		R&R				
	10	ADD BASE NUMBER	/		10						B 99999999	RO	RO		
	11	DIVIDE	/		11				RO	Q 5504	(POSSIBLE REMAINDER)				
	12	ADD QUOTIENT TO REMAINDER	6/		12					RO	5504				
	13	ADD A_2	6/		13				RO		18166 23670				
	14	R&R COUNTER TO GS4	/6		14						R&R				RI 23670
	15	EMIT 5 FROM EMITTER CONTROL	/		15					RI 5					
	16	MULTIPLY	/		16						A_3 118350				RO
THIRD ITERATION	17	R&R THIRD APPROXIMATE TO FS4	/2		17				A_3 11835		R&R				
	18	ADD BASE NUMBER	/		18						B 99999999	RO	RO		
	19	DIVIDE	/		19				RO	Q 8449	(POSSIBLE REMAINDER)				
	20	ADD QUOTIENT TO REMAINDER	6/		20					RO	8449				
	21	ADD A_3	6/		21				RO		11835 20284				
	22	R&R COUNTER TO GS4	/6		22						R&R				RI 20284
	23	EMIT 5 FROM EMITTER CONTROL	/		23					RI 5					
	24	MULTIPLY	/		24						A_4 101420				RO
FOURTH ITERATION	25	R&R FOURTH APPROXIMATE TO FS4	/2		25				A_4 10142		R&R				
	26	ADD BASE NUMBER	/		26						B 99999999	RO	RO		
	27	DIVIDE	/		27				RO	Q 9859	(POSSIBLE REMAINDER)				
	28	ADD QUOTIENT TO REMAINDER	6/		28					RO	9859				
	29	ADD A_4	6/		29				RO		10142 20001				
	30	R&R COUNTER TO GS4	/6		30						R&R				RI 20001
	31	EMIT 5 FROM EMITTER CONTROL	/		31					RI 5					
	32	MULTIPLY	/		32						SQUARE ROOT 100005				RO
	P	PUNCH	/		PCH	▨	▨	▨	▨	▨	SQUARE ROOT 10000				

Figure 11.15 Planning chart. Approximate square root.

However, the square root of 5,625 is developed in three iterations:

$$\frac{5{,}625}{30} + 30 \times 0.5 = 108 \quad \text{(second approximate)}$$

$$\frac{5{,}625}{108} + 108 \times 0.5 = 80 \quad \text{(third approximate)}$$

$$\frac{5{,}625}{80} + 80 \times 0.5 = 75 \quad \text{(square root of 5,625)}$$

Note that each iteration is a repetition of the previous step, with the difference that a new approximate is introduced as the divisor. Regardless of the number of iterations a base number may require, the machine always takes four.

The calculation requires 32 program steps, eight to each iteration. The size of the base number is determined on the read cycle so that on program 1 the digit 3, 30, 300, or 3,000 as the first approximate may be entered into FS 4.

The square root of a number having an odd number of digits to the right of the decimal point, e.g., 12.136, requires the addition of a zero to the right for accurate development of the root.

Gangpunching

Gangpunching may be done as a separate operation involving only the Card Read Punch or as part of a regular calculating run. Operations include straight or interspersed gangpunching, minor or major, and offset numerical or alphabetical. Digits may be emitted to punch a constant factor as all cards pass through the machine.

Questions and Exercises

1. Describe the various kinds of calculation that can be performed on the 604 Electronic Calculating Unit.
2. What is meant by "programmed" calculation?
3. How can calculation be machine checked?
4. How can procedure controls also be applied to calculating operations?
5. What are the three types of storage units in the 604?
6. List five of the specific operations that the 604 can perform (e.g., counter readout) and describe their functions.
7. What is the purpose of the planning chart and how is it used?
8. Plan the multiplication of $A \times B = P$. A is the unit price field from the stock room requisition prices as discussed in exercise 7, chapter 10. B is quantity issued from the same card. P is the total cost of the item.
9. Was the previous operation of gangpunching necessary in order to calculate total requisition cost in exercise 8?
10. Revise your stock room procedure to include the calculation of total requisition cost. Include appropriate controls.

11. Plan the calculation of $A \div B = Q$. A is the total cost of an inventory item that has been issued by requisitions over a period of time. B is the total number of pieces used. Q is the average cost of the item. The field A is in dollars and cents; B is a whole number; Q is to be calculated to the nearest cent.

REPORT PREPARATION 12

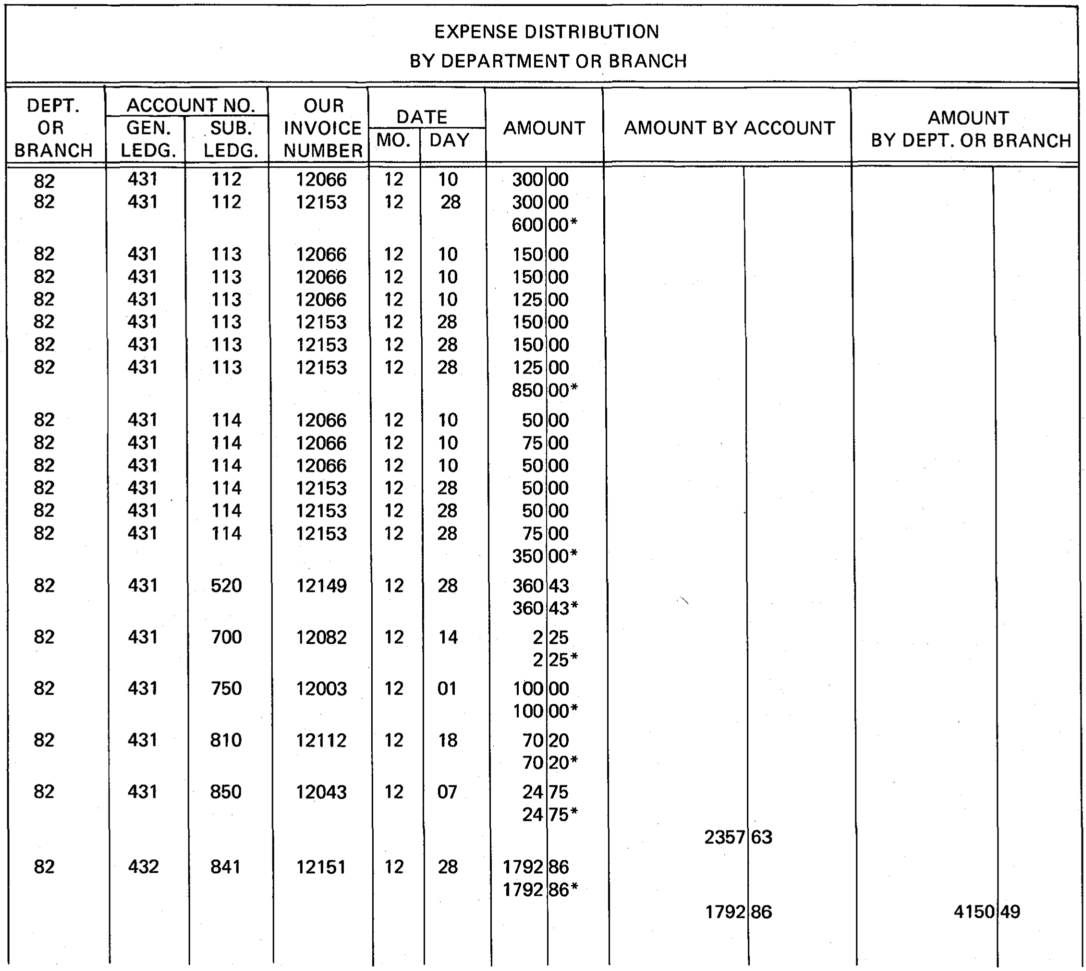
In punched-card data processing, printed documents or *reports* can be obtained as either intermediate or end results of a procedure. Many types and arrangements of reports can be produced automatically by various kinds of printing equipment known as *accounting machines*. This descriptive term is particularly appropriate because the machines combine a number of common accounting functions in a single unit. These include the capability to:

1. List or *detail print* data from each card in a file. Figure 12.1 is an example of how an expense distribution report can be detail printed. The cards from which the report is prepared have been sorted first by account number and then by department or branch number. The amount of each invoice, with date originated, is listed in account number sequence.

Like other punched-card equipment, accounting machines are equipped with removable control panels. The selection of data to appear on a given report can be determined by control panel wiring.

2. Tabulate or *summarize* the numerical data punched in a file of cards. For this purpose, the accounting machine has counters to add, subtract, multiply, and crossfoot totals.

Note that the report in Figure 12.1 is printed with totals *after each account*. This means that the machine not only accumulates amounts but recognizes any change in an identifying field, such as account number, between one card and another. This capability is used to control a total to print after each group has been listed. For example, in the report shown, two items are listed for account 431-112, six items for account 431-113, and five items for account 431-114. A total is printed after each group.



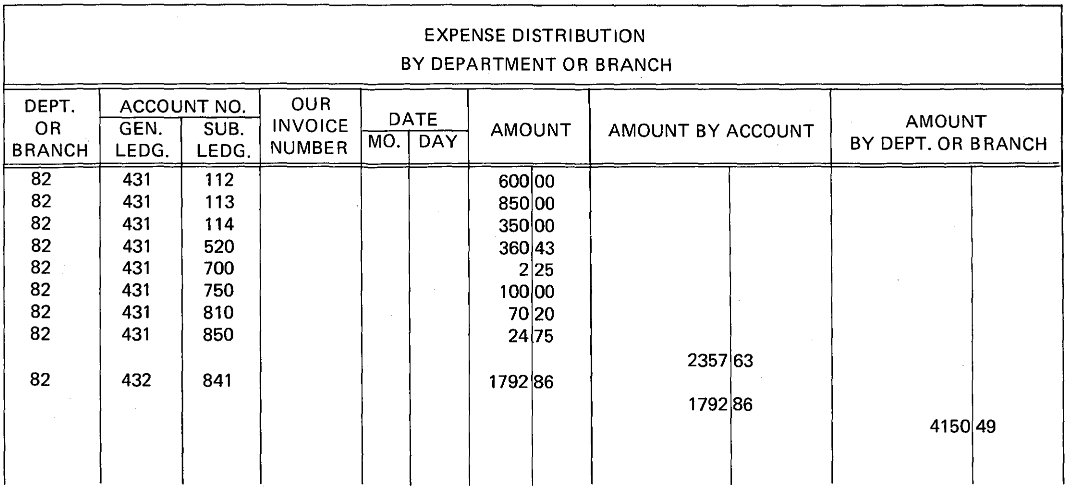
EXPENSE DISTRIBUTION
BY DEPARTMENT OR BRANCH

DEPT. OR BRANCH	ACCOUNT NO.		OUR INVOICE NUMBER	DATE		AMOUNT	AMOUNT BY ACCOUNT	AMOUNT BY DEPT. OR BRANCH
	GEN. LEDG.	SUB. LEDG.		MO.	DAY			
82	431	112	12066	12	10	300 00		
82	431	112	12153	12	28	300 00		
						600 00*		
82	431	113	12066	12	10	150 00		
82	431	113	12066	12	10	150 00		
82	431	113	12066	12	10	125 00		
82	431	113	12153	12	28	150 00		
82	431	113	12153	12	28	150 00		
82	431	113	12153	12	28	125 00		
						850 00*		
82	431	114	12066	12	10	50 00		
82	431	114	12066	12	10	75 00		
82	431	114	12066	12	10	50 00		
82	431	114	12153	12	28	50 00		
82	431	114	12153	12	28	50 00		
82	431	114	12153	12	28	75 00		
						350 00*		
82	431	520	12149	12	28	360 43		
						360 43*		
82	431	700	12082	12	14	2 25		
						2 25*		
82	431	750	12003	12	01	100 00		
						100 00*		
82	431	810	12112	12	18	70 20		
						70 20*		
82	431	850	12043	12	07	24 75		
						24 75*		
							2357 63	
82	432	841	12151	12	28	1792 86		
						1792 86*		
							1792 86	4150 49

Figure 12.1 Detail printing.

The next five accounts have one item each; therefore item and account totals are the same. All of these totals are printed whenever there is a change in any of the last three or *minor* digits of an account number.

Now notice that a different class of total appears after a change in the first three, or *intermediate*, digits of an account number. The intermediate total is the accumulation of all preceding minor totals. Also, at the end of the report, a *major* total is taken by branch or department number. An accounting machine can recognize up to three classes of control to print totals: minor, intermediate and major.



EXPENSE DISTRIBUTION
BY DEPARTMENT OR BRANCH

DEPT. OR BRANCH	ACCOUNT NO.		OUR INVOICE NUMBER	DATE		AMOUNT		AMOUNT BY ACCOUNT		AMOUNT BY DEPT. OR BRANCH	
	GEN. LEDG.	SUB. LEDG.		MO.	DAY						
82	431	112				600	00				
82	431	113				850	00				
82	431	114				350	00				
82	431	520				360	43				
82	431	700				2	25				
82	431	750				100	00				
82	431	810				70	20				
82	431	850				24	75				
								2357	63		
82	432	841				1792	86				
								1792	86		
										4150	49

Figure 12.2 Group printing.

Figure 12.2 shows that the same cards can be *group printed*. In this report, each individual card is not listed. Detail is omitted that shows how each total is accumulated. However, each total is identified with its corresponding account number. This form of control eliminates the effect of printing duplicate totals for single items.

3. Punch summarized data into cards when connected to a summary punch, for example, an IBM 519 Document Originating Machine. Figure 12.3 shows a summary card as it would be punched for the first line of the group-printed report. The amount ($300.00) is punched in columns 37 through 44 as an eight-position field. Branch and account numbers are punched in columns 13 through 20. Only the identifying fields and totals can be summary punched.

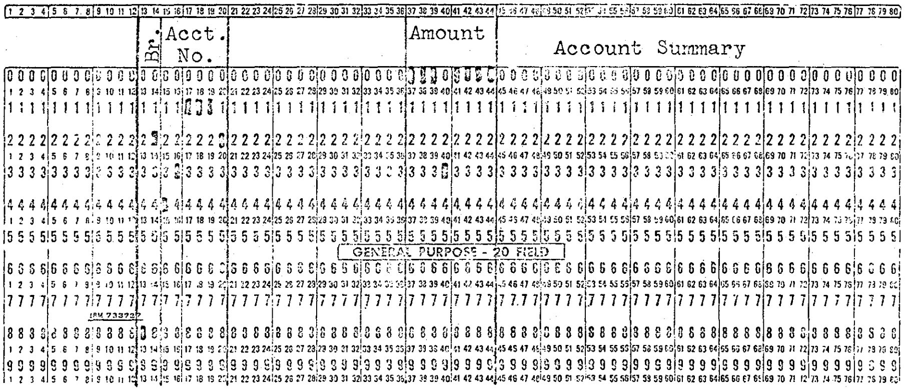


Figure 12.3 Summary card.

4. *Feed* and *position* continuous paper forms automatically to the line where the data should be printed. Spacing, form positioning, arrangement of the data on the form, total printing, and form-to-form ejection can all be controlled by a combination of panel wiring and the setup of an automatic carriage with punched paper tape.

Because printing can be done at speeds of over 100 lines per minute, the blank paper forms are usually fastened together as a continuous belt for machine feeding. After printing, individual forms are separated along perforated edges by special equipment.

Many documents produced by accounting machines are byproducts of processing within a procedure. Examples are invoices, statements, payroll checks, address labels, and so on. Figure 12.4 shows a continuous form invoice with matching paper tape for automatic carriage control.

Methods of Printing

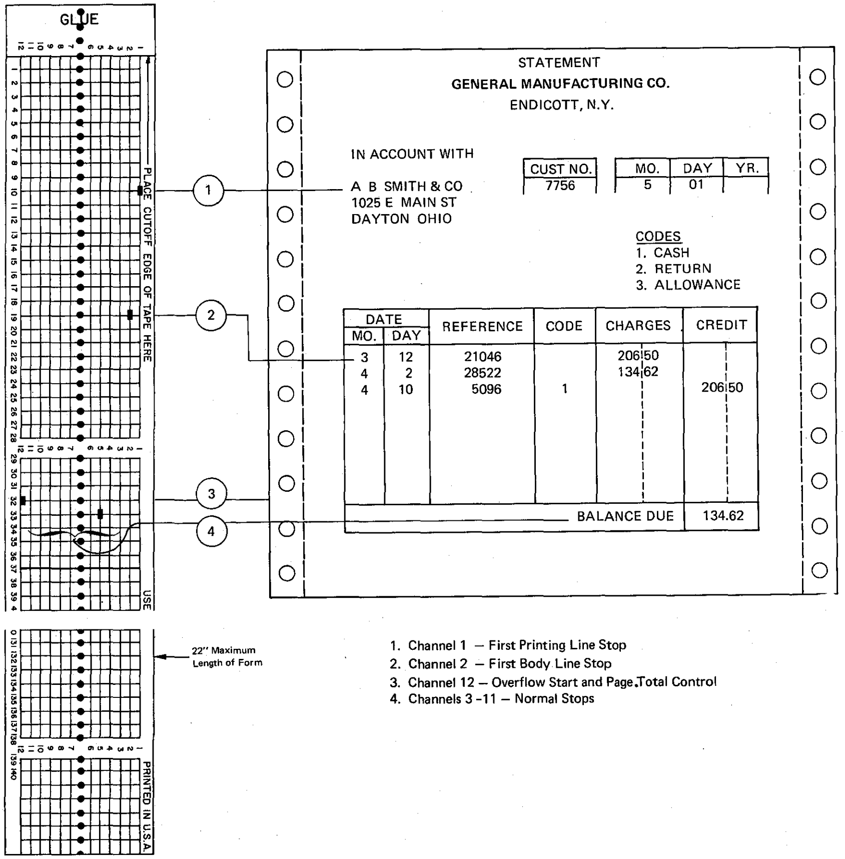
Printing is essentially a mechanical process, aided somewhat by the chemistry of ink transfer to paper by direct type impression.

For example, a typewriter is a mechanical printer, even though the mechanism in some models may be driven by electrical means. The typewriter is also a serial printer. That is, each character is individually chosen by the typist and printed, one at a time, in a line. After a character is printed, a carriage moves automatically to the left, thus positioning the paper properly to receive the next type impression. A different typebar is used for each character.

Punched card accounting machines are parallel printers, that is, an entire line of characters is put on paper in a single cycle. Complete lines are printed in parallel, one after the other, from the top of the form to the bottom. The printing mechanism is made up of a number of typebars or type wheels, each with a full set of characters. These may be numerical only; or they may be numerical, alphabetic, and special characters combined. The printing cycle is in two parts: a setup portion that positions the required character on each bar, and a very short impression portion when all the bars or wheels are pressed against the paper simultaneously. A carriage feeds the paper to the next printing line.

Like the typewriter, an inked ribbon is fed between the type and the paper to make a legible impression. The main advantage of parallel printing is speed.

Figure 12.4 Continuous form.



STATEMENT
GENERAL MANUFACTURING CO.
ENDICOTT, N.Y.

IN ACCOUNT WITH

A B SMITH & CO
1025 E MAIN ST
DAYTON OHIO

CUST NO.	MO.	DAY	YR.
7756	5	01	

CODES
1. CASH
2. RETURN
3. ALLOWANCE

DATE		REFERENCE	CODE	CHARGES	CREDIT
MO.	DAY				
3	12	21046		206 50	
4	2	28522		134 62	
4	10	5096	1		206 50
				BALANCE DUE	134.62

1. Channel 1 – First Printing Line Stop
2. Channel 2 – First Body Line Stop
3. Channel 12 – Overflow Start and Page Total Control
4. Channels 3 -11 – Normal Stops

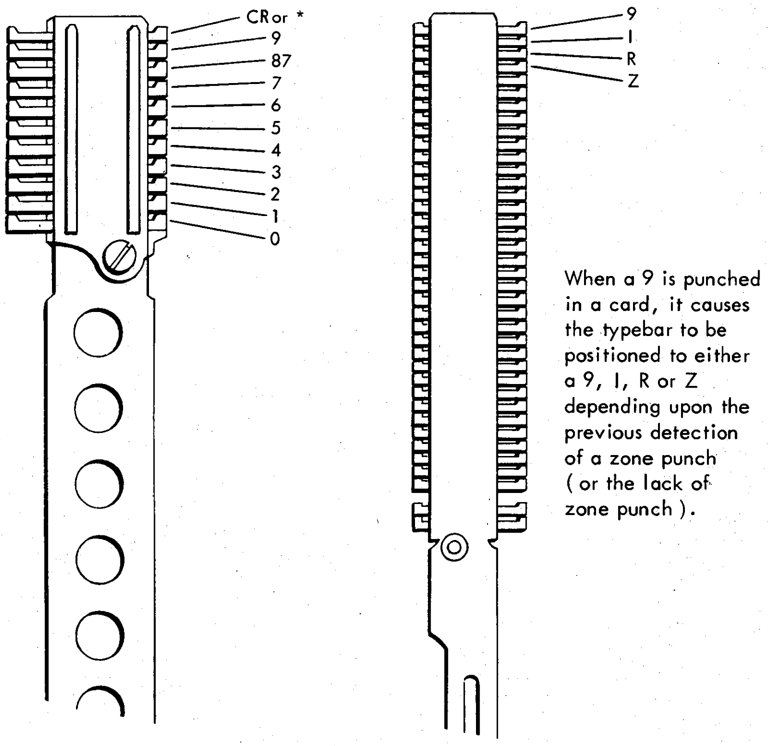
The following sections describe the basic operation of IBM accounting machines.

IBM 402 Accounting Machine

The IBM 402 Accounting Machine has two kinds of typebars in a single printing mechanism:

1. As many as 43 *alphanumeric*. Each typebar can print the 26 letters of the alphabet; the 10 numerical digits; and one special character, the ampersand (&).

2. As many as 45 numerical. Each typebar prints the 10 numerical digits and one symbol – either an asterisk (*) or a credit symbol (CR). The asterisk marks a plus total when printed on a report; the credit symbol marks a minus total or credit balance.



When a 9 is punched in a card, it causes the typebar to be positioned to either a 9, I, R or Z depending upon the previous detection of a zone punch (or the lack of zone punch).

Figure 12.5 Numerical type bar.

The two sets of bars are separated by a ribbon guide equivalent to one blank space. This space is an important consideration of form design.

As an amusing footnote, it is interesting to consider why the rather unusual number of 43 alphabetic type positions was chosen in the design of the machine. The first alphabetic printer marketed by IBM, the 405, also had the same arrangement of 43 bars. The reason is that it takes exactly 43 letter and space positions to spell out the name, International Business Machines Corporation. The original designers made sure their machine could print the full name of the parent company on a single line!

Figure 12.5 shows the type portion of a numerical and alphabetical typebar. Special characters and symbols can be specified as required in place of the standard arrangement. A certain number of removable bars can also be obtained to give the operator some flexibility in selecting characters. For example, extra-large digits of a special typeface may be desirable for printing the dollar amount on checks.

Card Reading

Figure 12.6 is a schematic of the card path through a 402 Accounting Machine. The feed hopper is located at the left of the machine where 80-column cards are placed by the operator, facedown, 9-edge toward the machine. Cards feed one at a time from the bottom of the hopper. Thus, when a file is in ascending sequence, cards feed from lower- to higher-order records in the normal sequence for report printing. After passing through the machine, all cards are automatically run into a stacker.

The schematic shows a *second* and *third* reading station. A first reading station, not shown here, is available only on another model (IBM 403) with the capability of printing multiple lines from a single card. The stations are fitted with 80 individual brushes, one for each of the 80 columns of the card. Cards pass between these wire brushes and a metal roller. Electrical contact is made whenever a brush touches the roller through a hole in a card. Rows of punched holes are sensed in parallel, from 9s at the bottom of the card to 12s at the top.

The two sets of brushes are spaced one after the other in the card path so that when a row of holes passes the third reading station, the same row in the following card is passing under the second station. For example, if row 8 is under the second reading station brushes, the following card is also at the 8 row position. This spacing arrangement is designed to permit a comparison to be made between data in two cards, an important function in sensing selected card fields for control purposes. Or, a control punch sensed at the second reading station can be used to prepare the machine to read, accumulate, or print selected data from that card when it reaches the third reading station.

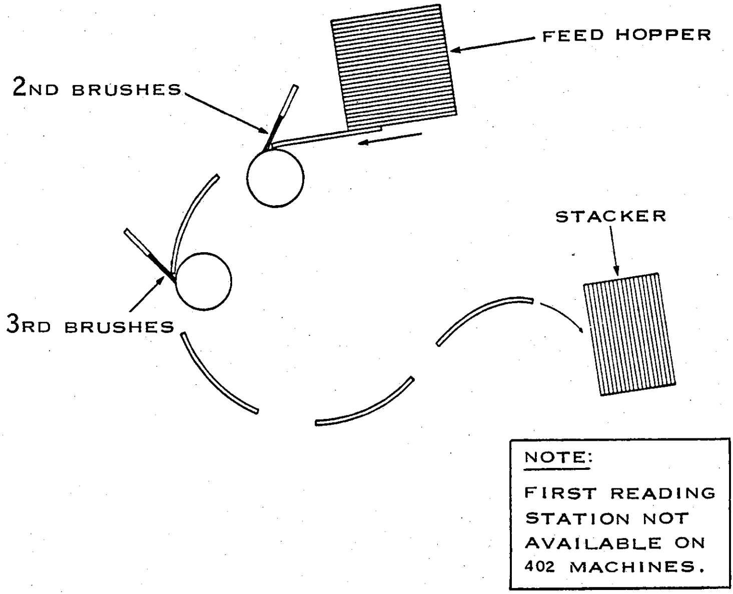


Figure 12.6 Card path, IBM 402 Accounting Machine.

Movement of cards past the reading stations is also timed precisely with the operation of other electromechanical devices in the machine: typebars, selectors, counters, and so on. Operation of the entire machine is in *cycles*, specific periods of time within which a given series of functions can be completed.

The 402 cycle is divided into 20 equal parts called points. Each point is further divided into 18 degrees. Thus, there are a full 360 degrees in every complete cycle. The time required for any function can be expressed in degrees. For example, card reading occurs every cycle between 0 and 216 degrees. The remainder of the cycle to 360 degrees represents the space between cards as they pass through the machine. Cycles are repeated, one after the other, as long as the machine is running. All functions are timed to operate in an exact relationship to each other within a given cycle. Overall machine capability can be adapted to a wide variety of procedures by control panel wiring.

Control Panel

By control panel wiring, input data to be printed is selected from punched cards as they are fed past the two reading stations. The data, in the form of timed electrical pulses, is directed to specific devices within the machine. The devices are also controlled by wiring to perform different functions, depending upon the requirements and format of the report to be printed.

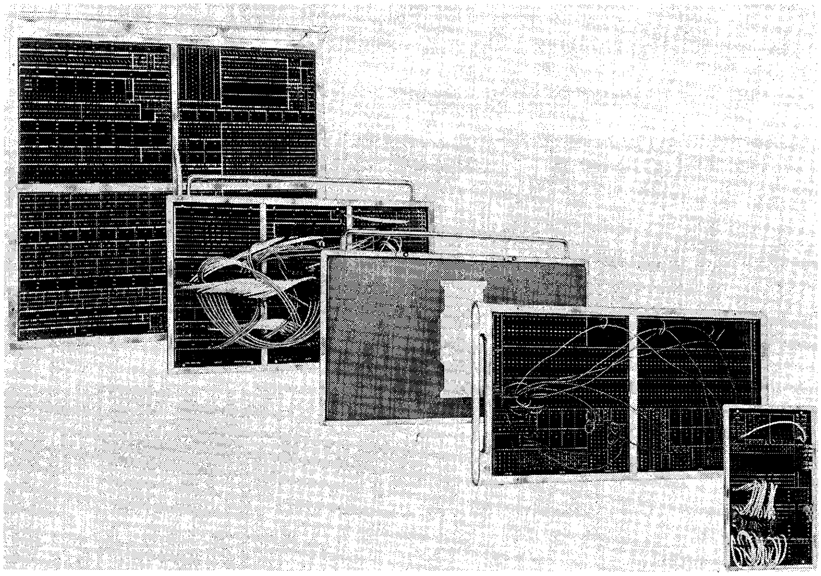


Figure 12.7 IBM accounting machine control panels.

Control panels are constructed of special insulating material held in place by metal frames. Figure 12.7 shows several types of control panel for IBM punched-card equipment, including the 402. The 402 panel is made up of three sections, each drilled with as many as 34 rows of holes. Figure 12.8 is a schematic of this arrangement. Rows are marked A through HH from top to bottom; holes are numbered 1 through 66 from left to right. The labeling system makes it possible to identify any hole by referring to the intersection of the row and hole locations. For example, *card cycle* holes are located at rows J through Q and numbers 49 and 50. The various machine functions of the holes are printed on the surface of the control panel.

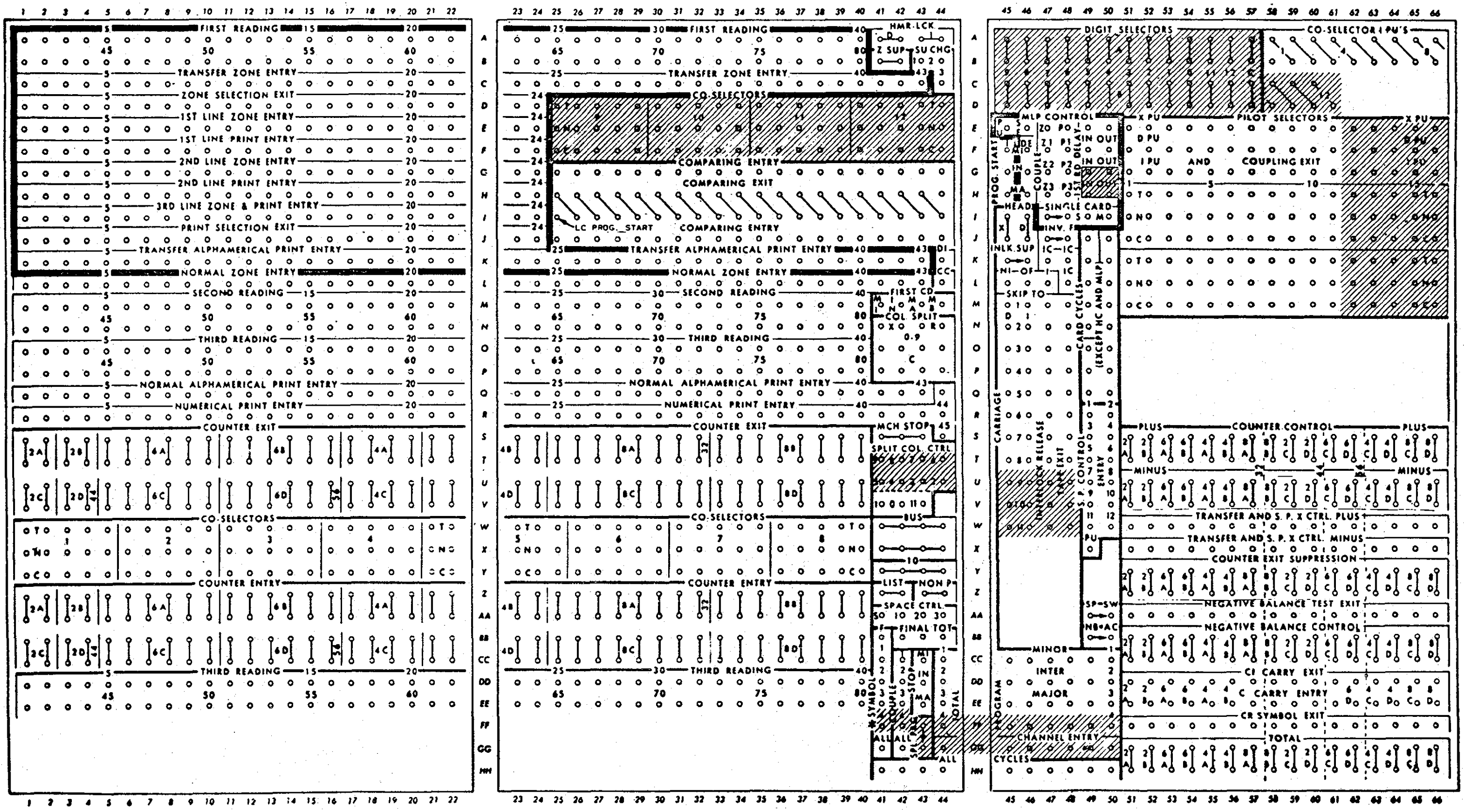


Figure 12.8 Schematic. IBM 402 Accounting Machine control panel.

Each hole is fitted with a metal prong in the back which serves as a retainer for a *plugwire*. A plugwire is inserted manually from the front of the control panel. The wires are insulated, except at each end, where a small jackplug holds the wire in place. Thus, holes or *hubs* in the control panel can be connected together with the pluggable wiring. The 402 control panel shown in Figure 12.7 has plugwires in place.

The control panel is inserted in the machine by means of a metal slide that clamps firmly in place. Spring contacts on the machine connect with the metal prongs on the back of the control panel, thus completing an electrical circuit between the internal wiring of the machine and the external wiring of the control panel. For example, when contact is made between a brush and the metal roller at a card reading station, the resulting impulse is directed by internal wires to the control panel, from where it is redirected to some machine device. The device is also connected to the panel. Figure 12.9 illustrates this arrangement schematically.

Each panel, then, must have two kinds of hub: *exits* and *entries*. An exit hub is one which emits an impulse. Some exits are under the control of the hole in the card; others may result from some function previously performed, or may be automatic for every card. An entry hub is one which can accept an impulse wired to it, for example, a counter entry. A connection must always be made from an exit to an entry by placing one end of a plugwire in the exit hub and the other end in the entry hub. Which exits and entries are used will depend entirely upon what job a machine is called upon to do.

Whenever two or more hubs are connected together by lines (Figure 12.10), these hubs are *common*, that is, two or more exits and entries serve the same purpose. An arrow between two hubs identifies them as a switch that can be turned on by connecting the two hubs. A curved arrow indicates the switch can be connected through a selector.

Control panel wiring is changed to prepare each type of report, thereby giving one machine the general purpose flexibility to produce a number of different types of documents or reports for many different applications.

Once wired, a panel can be stored indefinitely. This means that whenever a required report is to be printed, the same panel can be used again and again. A panel can also be wired *permanently* with a special type of plugwire furnished especially for this purpose. The panel is then fitted with a metal cover for storage

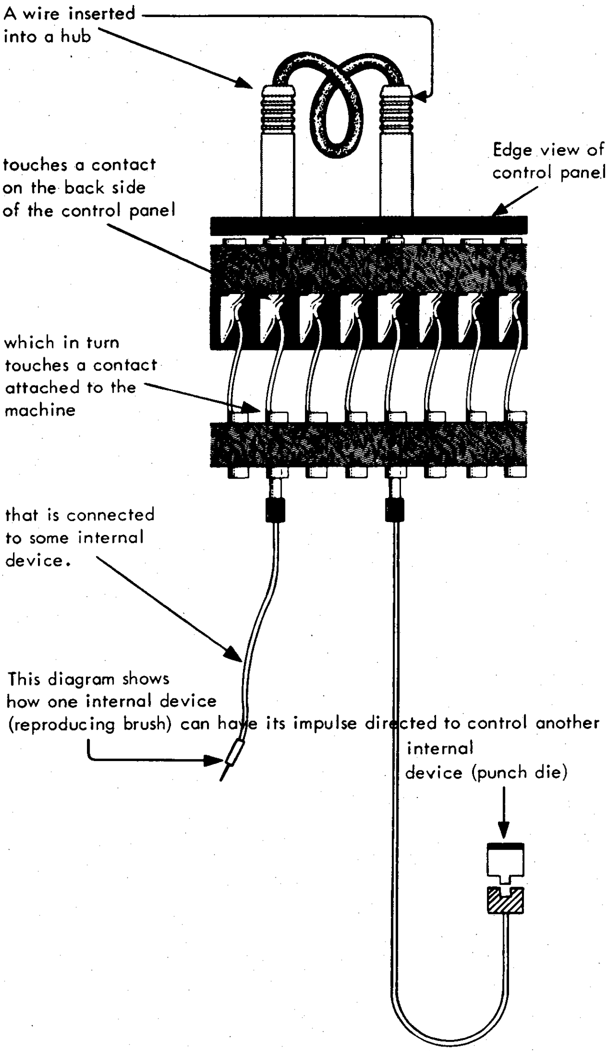

A wire inserted into a hub
touches a contact on the back side of the control panel
Edge view of control panel
which in turn touches a contact attached to the machine
that is connected to some internal device.
This diagram shows how one internal device (reproducing brush) can have its impulse directed to control another internal device (punch die)

Figure 12.9 Schematic. Control panel wiring to internal devices.

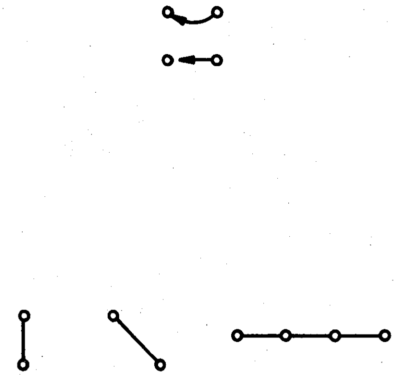


Figure 12.10 Control panel common hubs.

(see Figure 12.7). Thus, any number of panels can be kept on hand to correspond with the variety of report printing requirements called for by a particular procedure.

Control Panel Wiring Any detailed presentation of control panel wiring is considerably beyond the intended scope of this text. Complete information on this subject is readily available in reference manuals prepared and distributed by the manufacturers of punched-card equipment. However, a few examples are presented in the following sections to illustrate some basic operational principles of these extremely versatile machines. These principles are more or less common to all IBM punched-card machines having control panels. The examples are intended to provide background for use in the design and study of data processing procedures.

The 402 Accounting Machine is chosen for these examples because it is about the simplest model of this type of equipment now available. Later sections include descriptions of some of the features of the more advanced 407.

As previously stated, the 402 can be equipped with as many as 88 typebars – 43 *alphamerical* and 45 *numerical*. Each alphamerical typebar has all letters of the alphabet, all digits 0 through 9, and one special character position with an ampersand (&). Alphamerical typebars are positioned to the left side of the print area. Each numerical typebar has 10 numerals and one symbol. In odd-numbered typebars the symbol is an asterisk (*), and in even-numbered typebars it is a credit symbol (CR). From this arrangement, it follows that alphabetic information can be printed only on the left side of the report, while numerical information can be placed in any location. Normally, this restriction is of little consequence since conventional report design places the name or description to the left side with quantities and amounts to the right.

As an aid to good form design, it is often helpful to use a ruled layout sheet. An example is shown in Figure 12.11. All accounting machine typebar positions are spaced and numbered, just as they are positioned in the machine. Sheets can be obtained full size, so that a finished layout will be actual size. The completed sheet is also an aid to control panel wiring, and can be used for ordering forms of proper size and specifications from the forms manufacturer.

Figure 12.12 shows a punched-card source document from which a cash requirements statement is to be run. From the layout in the preceding illustration, it can be seen that the form has been designed to print in alphamerical typebars 24 through 43 and numerical typebars 1 through 27. Vendor abbreviation must print from card columns 7 through 17 into alphamerical typebars 24 through 34. Vendor number must print from columns 18 through 22 into typebars 35 through 39. Invoice amount is printed from

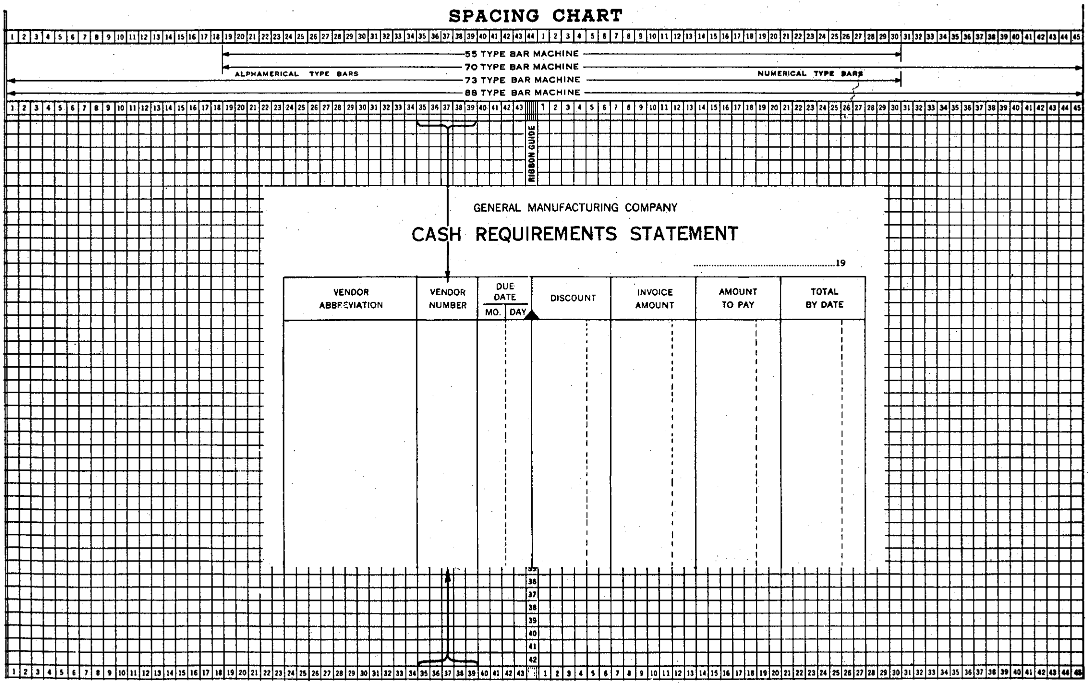


Figure 12.11 Accounting machine form layout sheet.

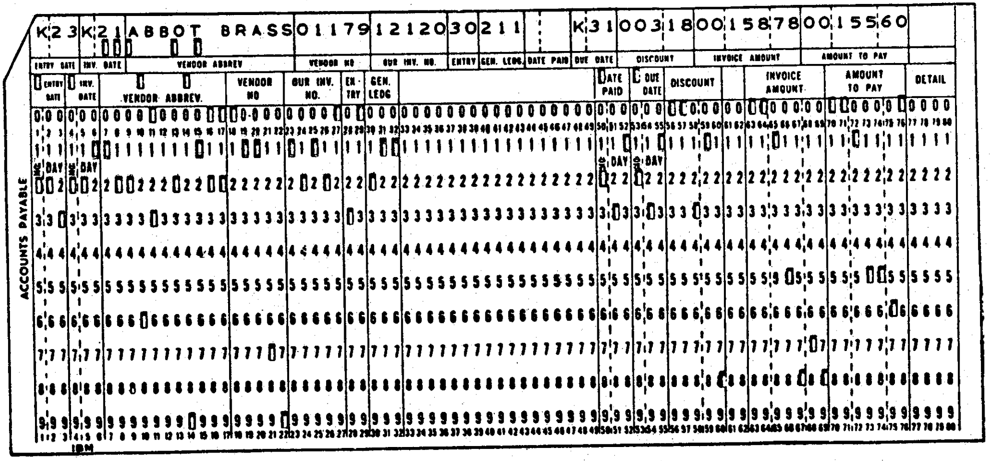
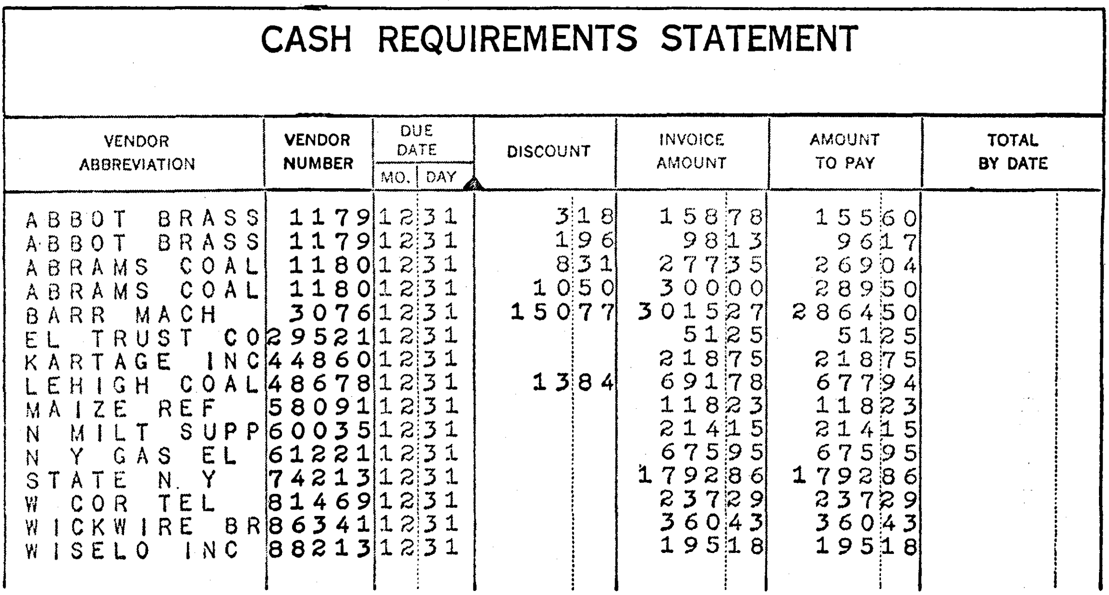


Figure 12.12 Accounts payable source card.

columns 63 through 69 into numerical typebars 7 through 13, and so on. Notice that the space between 43 and 1 represents the ribbon guide to divide the alphamerical typebars from the numerical typebars.



CASH REQUIREMENTS STATEMENT

VENDOR ABBREVIATION	VENDOR NUMBER	DUE DATE		DISCOUNT	INVOICE AMOUNT	AMOUNT TO PAY	TOTAL BY DATE
		MO.	DAY				
ABBOT BRASS	1179	12	31	318	15878	15560	
ABBOT BRASS	1179	12	31	196	9813	9617	
ABRAMS COAL	1180	12	31	831	27735	26904	
ABRAMS COAL	1180	12	31	1050	30000	28950	
BARR MACH	3076	12	31	15077	301527	286450	
EL TRUST CO	29521	12	31		5125	5125	
KARTAGE INC	44860	12	31		21875	21875	
LEHIGH COAL	48678	12	31	1384	69178	67794	
MAIZE REF	58091	12	31		11823	11823	
N MILT SUPP	60035	12	31		21415	21415	
N Y GAS EL	61221	12	31		67595	67595	
STATE N. Y	74213	12	31		179286	179286	
W COR TEL	81469	12	31		23729	23729	
WICKWIRE BR	86341	12	31		36043	36043	
WISELO INC	88213	12	31		19518	19518	

Figure 12.13 Cash requirements statements.

The report is to be detail printed. This means that data from all cards in the file will be listed on the finished report as shown in Figure 12.13. The required panel wiring is described next.

Detail Printing

When detail printing, the operation of the machine is controlled in such a way that the typebars are actuated to print during every read cycle. Two sets of control panel hubs are available to do this (refer to Figure 12.8).

All Cycles

These hubs, located in HH, 41-50, emit impulses that can be used to control many functions of the machine. The emitted impulses are available every machine cycle.

List

When the list hubs are impulsed from exit hubs, all 88 typebars of the machine rise as each card passes the third reading station. If there are holes in the card and the control panel is properly wired, the typebars print every time they rise. Detail printing thus occurs at a rate of 50 or 150 lines per minute, depending upon the model. Spacing of the paper by the carriage is automatic before each line prints. If every card is to be printed, the list hubs are wired directly from all cycles. The list hubs are located at Z, 41-42.

Normal Alphamerical Print Entry

Each alphamerical typebar has a corresponding normal print entry hub on the control panel. These hubs are located in row Q, 1-43. When the hubs are wired from the third reading brushes, the typebars print the *numerical* information punched in the corresponding columns of the card passing that station. Therefore, to print vendor number on the cash requirements statement, card columns 18 through 22 are wired from third reading to alphamerical normal print entry bars 35-39. A diagram of this wiring is shown in Figure 12.14.

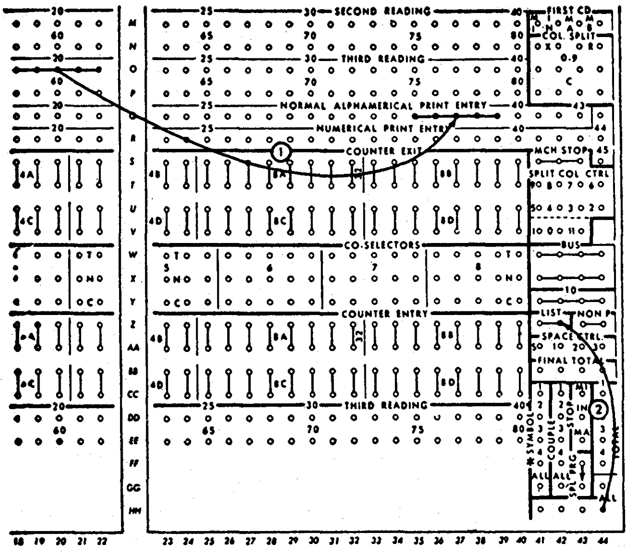


Figure 12.14 Detail printing. Numerical.

In detail printing, the upward movement of all typebars is synchronized with card reading. The bars move up while a card is being read from the 9 to the 12 row. As soon as a brush senses a punched hole, the typebar is latched at that position. For example, if a 4 is read, the typebar wired to that column stops moving upward when the digit 4 is in the printing position. The typebar is held in this position during the remainder of the reading cycle. After the brushes have read all of the punching rows 9 through 12, all typebars will have been properly positioned for printing. Each typebar has a hammer which, when fired, pushes the printing type against the carriage platen. All the hammers are fired at one time to print an entire line.

Third Reading

The third reading brushes are used for all normal reading operations. Impulses from these exit hubs must be directed either to normal alphamerical or numerical print entry hubs.

There are two sets of third reading brush hubs on the control panel. One set is located in rows O-P, 1-40, and the other is located in rows DD-EE, holes 1-40. The corresponding hubs in both sets are common and can be wired interchangeably. The two sets are needed when it is necessary to wire the same card columns into the typebars for listing and at the same time accumulate the amounts being printed in counters.

Numerical Print Entry

Each numerical typebar also has a corresponding print entry hub on the control panel. The hubs are located in row R, holes 1-44. The hub for print entry 45 is located just below 44. When a numerical print entry hub is impulsed from a third reading brush hub, the typebar prints the numerical information punched in the corresponding column of the card. To print the amount to pay on the cash requirements statement, third reading hubs 70-76 are wired to numerical print entry hubs 14-20. Wiring for the other listed items of numerical information is done in the same manner.

Second Reading

The 80 second reading hubs are outlets for another set of 80 brushes located just above the third reading brushes. These hubs are located in M-N, 1-40. Like the third reading brushes, they read all 80 columns of the card. However, they are used primarily on the 402 to prepare the machine to print letters instead of numbers. They may also be used to perform other specific functions which will not be covered here.

Normal Zone Entry

To print an alphabetic character, the typebar must receive a zone impulse on one cycle and the corresponding digit impulse on the next cycle. Zone punches are read from the second reading brushes and are accepted by any of the 43 normal zone entry hubs. A zone punch positions the typebar one cycle in advance so that the proper character will print on the following cycle when the digit punch is read.

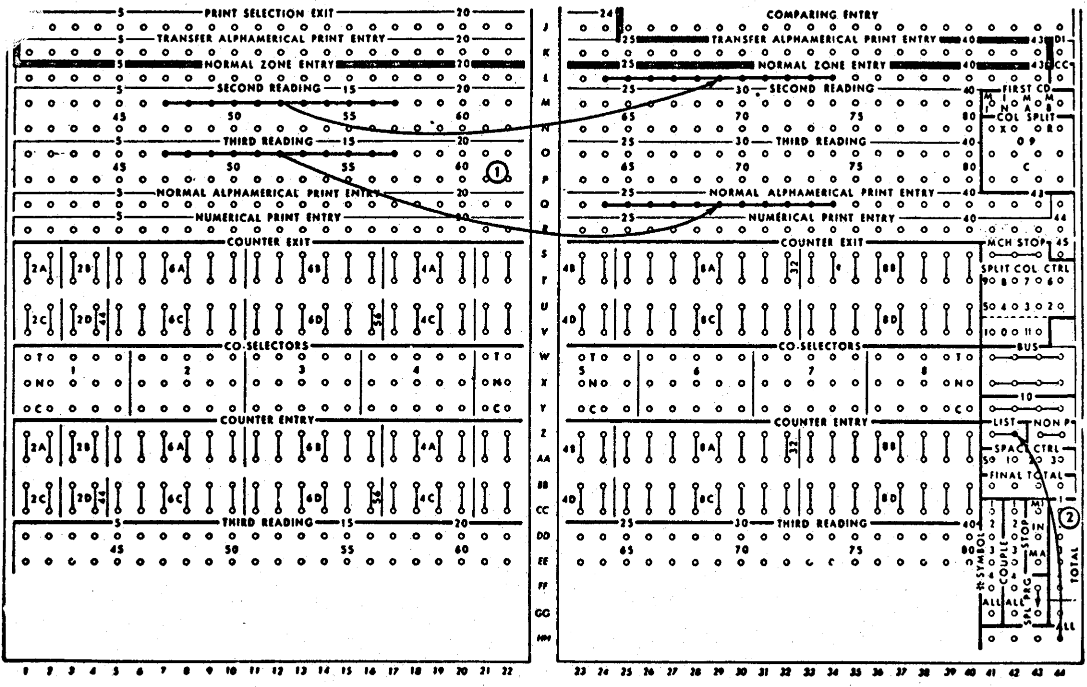


Figure 12.15 Detail printing. Alphabetic.

The cash requirements statement shown in Figure 12.13 has one column of alphabetical information headed Vendor Abbreviation. The item is printed in typebar positions 24 through 34. Therefore, card columns 7 through 17 are wired from second reading to normal zone entry hubs 24 through 34; card columns 7 through 17 are also wired from third reading to normal alphamerical print entry hubs 24 through 34. The typebars receive the zone signals when the cards pass the second reading station; the proper digit signals when the cards pass the third reading station. The wiring diagram is shown in Figure 12.15.

This example further illustrates the flexibility of report format that can be produced by the 402 Accounting Machine. Data read from cards can be printed in almost any arrangement, provided alphabetic information is kept within the 43 positions to the left of the printing mechanism. Notice that printing speed remains constant regardless of the amount of information contained in the report. However, the time between printing cycles may be extended slightly by paper form positioning and ejection, depending upon the length of the form and the position of heading and total lines.

Accumulating Totals

To perform the arithmetic operations of addition and subtraction, accounting machines are equipped with counters. Each counter is made up of single-position wheels or *accumulators* grouped into units of 2, 4, 6, and 8 positions each. The number of counters available depends upon the model. A 402 can be equipped with as many as four 2-position counters, four 4-position, four 6-position, and four 8-position counters for a total of 80 positions. The result is one of having available a number of individual adding machines that can be used to accumulate totals.

Each counter in the machine is identified by a number and a letter. The number indicates how many positions the counter has and the letters A, B, C, and D identify the specific counter of the group. Two-position counters are labeled 2A, 2B, 2C, and 2D; 4-position counters are labeled 4A, 4B, 4C, and 4D.

Counters may be coupled together in any desired combination. An 8-position may be coupled with a 2-position to form a 10-position counter. Or, a 6-position and a 4-position may be coupled to form a 10-position counter. Two 4-position counters may be coupled to form one 8-position counter. Thus, the size of the counter may be adjusted to the expected number of digits in a report total.

To accumulate totals, four basic wiring steps are required:

1. Determine what card fields are to be accumulated. These are normally amount or quantity fields read at the third reading station. Card columns are wired to a counter entry.

2. Select the cards to be added or subtracted. Accumulation can be restricted to specifically designated cards in a file, provided these cards are punched with a unique identifying hole. For example, when a file contains both master and detail cards, all cards pass the reading stations. But the counters can be controlled to accumulate only from details and not from masters. The master cards are ignored by wiring proper exit hubs to counter control hubs.

Also, by sensing selected control punches, one class of card can be controlled to add, while a second class of card in the same file can be controlled to subtract. This capability is required in nearly all types of accounting applications. For example, in an accounts payable file, purchases add while payments to vendors subtract. In a payroll file, employee earnings add; deductions subtract.

The selected impulses are wired to a counter control hub.

3. Determine when a total is to be printed. There are four classes of totals that can be printed on a report: minor, intermediate, major, and final. These levels suffice for nearly all quantity or amount totals required in commercial accounting practices.

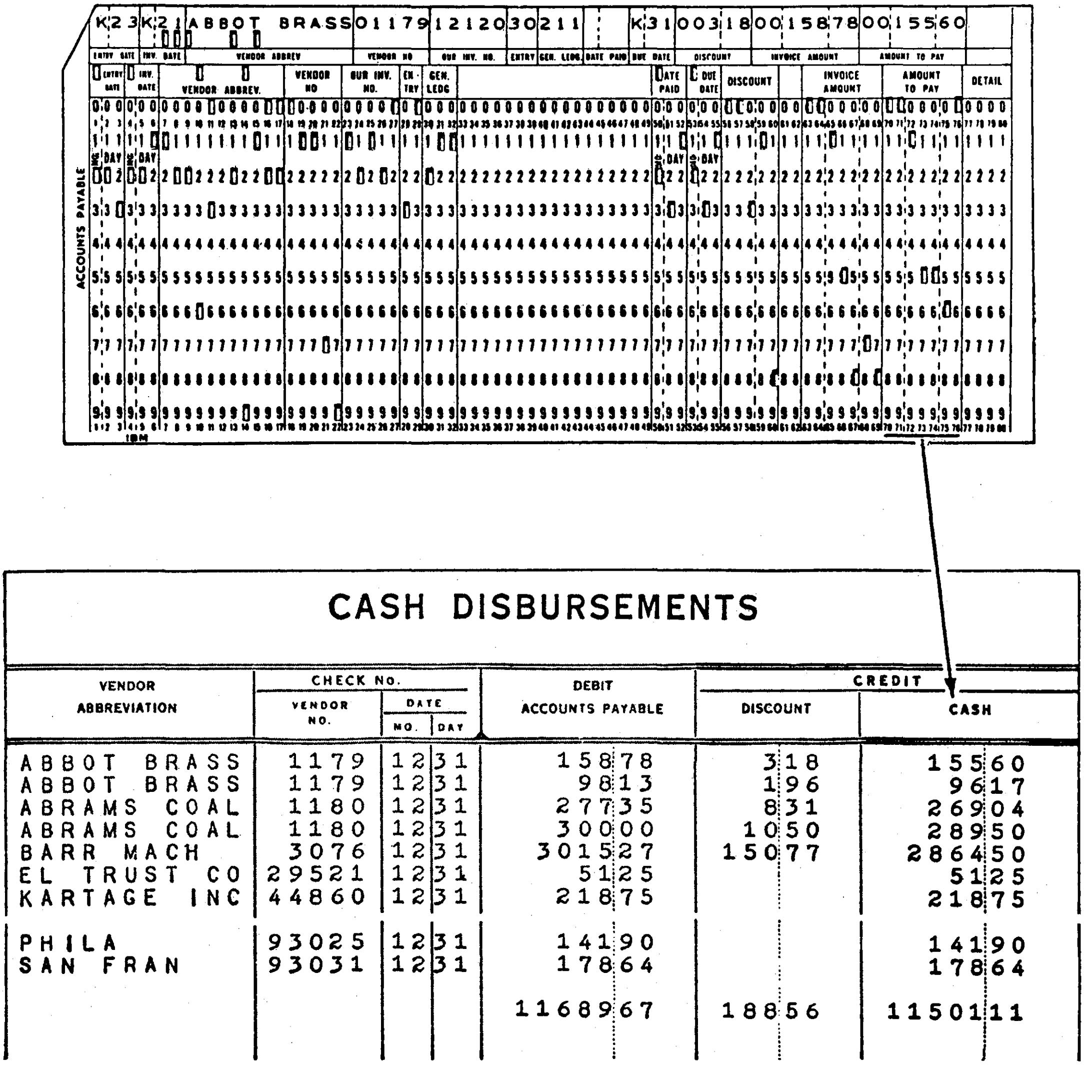
A minor total is printed at the lowest level of accumulation, e.g., by item, employee, or account number. The intermediate total is an accumulation of two or more minor totals: all items for a customer, all employees in a department, all accounts in a ledger. The major total represents an accumulation of all minor and intermediate totals: all customers in a territory, all departments in a factory, all ledgers in a closing statement.

By control panel wiring, counters in the accounting machine can be controlled to print at any of the above levels. The counter normally resets as printing occurs.

The final or grand total is the end accumulation of all other totals, that is, the final accounts receivable total, final complete payroll, and so on.

4. Determine where a total is to be printed. Counter exit hubs are available to cause typebars to print during the total cycle. Totals may be printed in any typebar positions. However, it is usually desirable for totals to appear in the same typebar position as the listed item quantities or amounts being accumulated.

Figure 12.16 shows a cash disbursements report to be listed from the accounts payable card form also shown in the illustration.


K23 K21 ABBOT BRASS 01179 12120 30211 K31 00318 0015878 0015560
ENTRY DATE
INV. DATE
VENDOR ABBREV
VENDOR NO
OUR INV. NO.
ENTRY
GEN. LEDG
DATE PAID
DUE DATE
DISCOUNT
INVOICE AMOUNT
AMOUNT TO PAY
DETAIL
ACCOUNTS PAYABLE
IBM

CASH DISBURSEMENTS

VENDOR ABBREVIATION	CHECK NO. VENDOR NO.	DATE MO.	DATE DAY	DEBIT ACCOUNTS PAYABLE	CREDIT DISCOUNT	CREDIT CASH
ABBOT BRASS	1179	12	31	158 78	3 18	155 60
ABBOT BRASS	1179	12	31	98 13	1 96	96 17
ABRAMS COAL	1180	12	31	277 35	8 31	269 04
ABRAMS COAL	1180	12	31	300 00	10 50	289 50
BARR MACH	3076	12	31	3015 27	150 77	2864 50
EL TRUST CO	29521	12	31	51 25		51 25
KARTAGE INC	44860	12	31	218 75		218 75
PHILA	93025	12	31	141 90		141 90
SAN FRAN	93031	12	31	178 64		178 64
				11689 67	188 56	11501 11

Figure 12.16 Accumulating a final total.

Columns 70 through 76 are to be accumulated and printed as a final total after all cards have been listed. For purposes of illustration, it is assumed that no other classes of totals are required. Figure 12.17 is a schematic of the control panel wiring needed to list and total the amount to pay field.

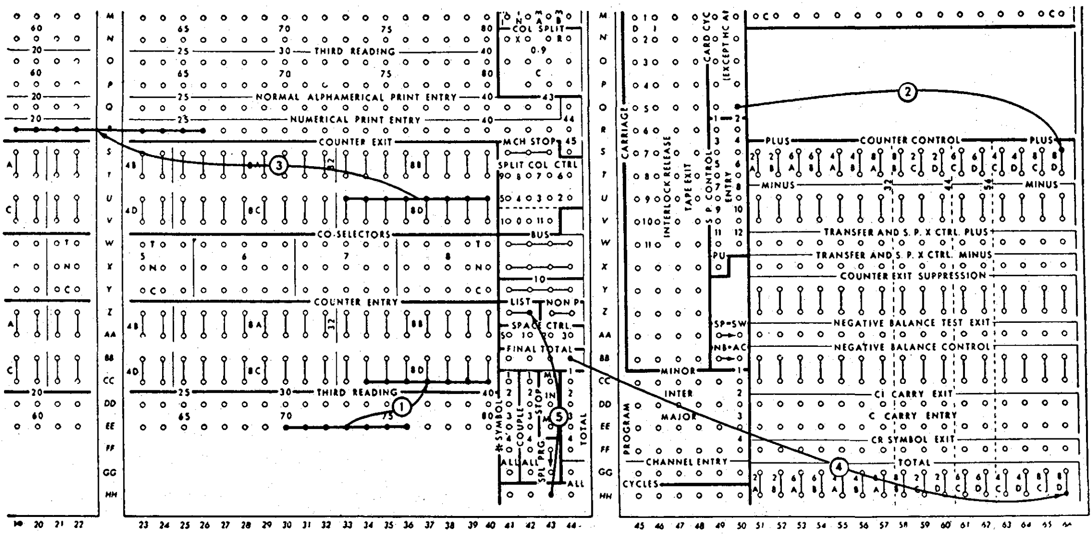


Figure 12.17 Panel wiring. Final total.

1. The amount to pay field is wired from the third reading brushes (EE, 70-76) to counter entry 8D (UV, 33-40). The lower set of third reading hubs is used since it is nearest the counter entry. Note that each counter in the machine has a corresponding set of counter entry hubs on the control panel.

The size of the counter used must be predetermined so that enough positions will be available to print the anticipated number of digits in the total. This can only be done by knowing the application for which the report is to be printed. In this case, any 8-position counter or any combination of counters can be used that total eight positions, such as 6B and 2B or 4A and 4B.

2. Counter 8D is impulsed to add from a card cycles hub. These hubs emit an impulse as each card passes the third reading station. In this example, it is assumed that all cards are to be added.

3. The amount to pay (cash) is detail printed by wiring from the 8D exit hubs to the numerical print entry, positions 19 through

26. The same printing could be accomplished by wiring from the third reading brushes (P, 70-76) to numerical print entry hubs 19-26.

4. The total accumulated in 8D is printed at the end of the run. The counter is cleared or reset to zero by wiring a final total exit (BB, 41-44) to 8D total (HH, 66). Each counter in the machine has a total control hub on the panel (GG, HH, 51-65).

The final total exit hub emits an impulse that is available when the card hopper is empty, the machine is idling, and both the final total and start keys are pressed simultaneously by the operator. Thus, a final total can only be taken intentionally after all cards have passed through the machine. In this example, manual spacing of the carriage would be necessary before printing the final total.

5. All cycles, wired to list, cause the machine to print every card.

Program Control

An accounting machine can be set up to print totals automatically at the minor, intermediate, and major levels by proper control panel wiring of a comparing unit. These hubs are located on the 402 panel at G-J, 25-44 as shown in Figure 12.18. The comparing unit is divided into individual positions with each position available to compare one card column. A position has two comparing entry hubs and two comparing exit hubs. The total number of positions is dependent upon the machine model but at least 10 are furnished as standard equipment on all models.

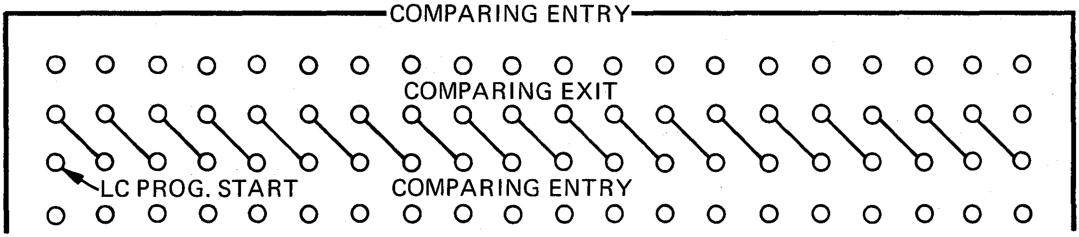
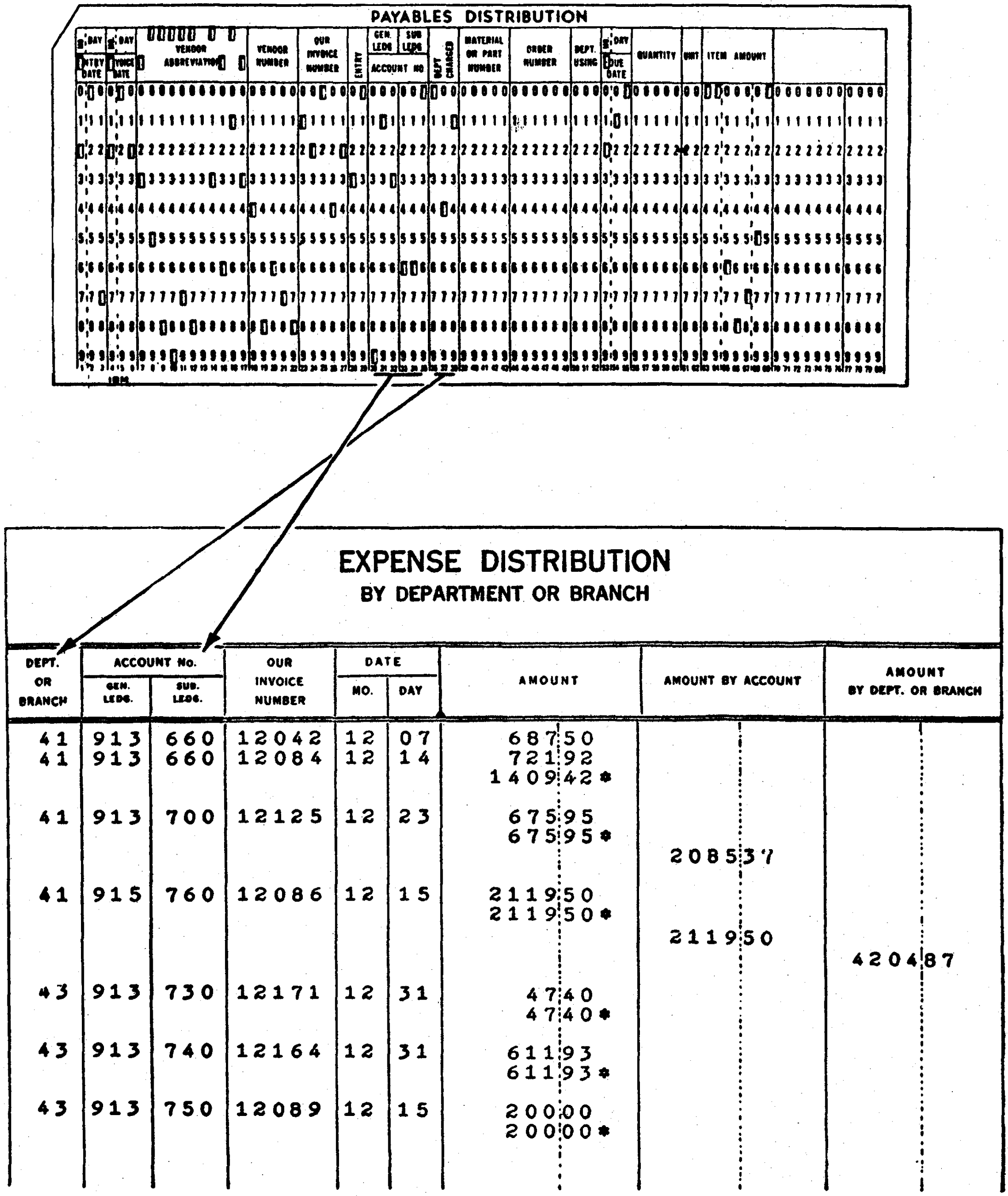


Figure 12.18 Comparing unit.

From the second and third sets of reading brushes, a field can be read from two successive cards at the same time. When wired to the comparing unit, any given field in a card passing the second brushes can be compared with the same field in a card passing the third brushes.

Comparison is made by wiring the field from the second reading hubs to one row of comparing entries, and from the third



EXPENSE DISTRIBUTION
BY DEPARTMENT OR BRANCH

DEPT. OR BRANCH	ACCOUNT No.		OUR INVOICE NUMBER	DATE		AMOUNT	AMOUNT BY ACCOUNT	AMOUNT BY DEPT. OR BRANCH
	GEN. LEDG.	SUB. LEDG.		MO.	DAY			
41	913	660	12042	12	07	687 50		
41	913	660	12084	12	14	721 92		
						1409 42 *		
41	913	700	12125	12	23	675 95		
						675 95 *		
							2085 37	
41	915	760	12086	12	15	2119 50		
						2119 50 *		
							2119 50	
								4204 87
43	913	730	12171	12	31	47 40		
						47 40 *		
43	913	740	12164	12	31	611 93		
						611 93 *		
43	913	750	12089	12	15	200 00		
						200 00 *		

Figure 12.19 Expense distribution report and Accounts Payable distribution card.

reading hubs to the other row of comparing entries. Either row of comparing entries may be wired from either set of brushes and any positions may be used. However, the field must be wired to the same positions from each set of brushes. For example, if columns 10 through 15 are wired from second reading to comparing positions G, 30-35, columns 10 through 15 must also be wired from third reading to comparing positions J, 30-35.

If the fields being compared are identical, no impulses are available at the comparing exit hubs. When the punching in one card does *not* compare with the punching in the preceding card, impulses *are* available at these hubs. A comparing exit hub is wired to a program start hub (FH, 45), either to minor, intermediate, or major, depending upon which level total is to be printed. Since it cannot be predetermined which column of the compared field will cause an exit impulse, the comparing exit hubs are connected together with a single exit wired from either the left or right end column of the field. The following example explains numerical program control wiring in greater detail.

Figure 12.19 shows an expense distribution report detail printed from a payables distribution card. Three levels of totals are taken on the item amount field punched in columns 63 through 69. These three levels are:

1. Minor total by subledger. The first minor total, $1,409.42, is printed from one counter when the 402 comparing unit senses a change from 660 to 700 in the subledger number.

2. Intermediate total by general ledger number. The first intermediate total, $2,085.37, is printed from a second counter when the general ledger number changes from 913 to 915.

3. Major total by department or branch. The first major total, $4,204.87, is printed from a third counter when the department number changes from 41 to 43.

Program Start

There are three program start hubs (F-H, 45) labeled minor, intermediate, and major. These hubs receive comparing exit impulses to stop card feeding and start a series of program total cycles: one program for minor, two for intermediate, and three for major. If intermediate program start is wired alone, a minor total cycle is forced before the intermediate total cycle. If major program start is wired alone, both a minor and an intermediate total cycle are forced before the major total cycle.

Total Program

Each program level has seven exit hubs which emit all cycle impulses whenever the corresponding program start hub is impulsed (CC-FF, 44-50). Program start must be impulsed before these hubs become active. Minor program exits emit impulses when the minor program start is impulsed. Minor and intermediate program exits emit impulses when the intermediate program start is impulsed. Minor, intermediate, and major program exits emit impulses only when the major program start is impulsed.

Counters read out and reset automatically when a total program is wired to a counter total entry. After total printing the machine restarts automatically.

Before the payables distribution file can be fed through the accounting machine, all cards must be sorted numerically, first on columns 30 through 35 and then on columns 36 through 38. Figure 12.20 shows the control panel wiring for the expense distribution report.

1. The subledger field is wired from columns 33 through 35 of both second and third reading to the comparing entry. Corresponding comparing exits are wired together to make all three positions common. If there is an unequal condition in any of the three columns of the field, a comparing exit impulse will be available. This impulse is directed to program start minor. As a result, the machine begins a program cycle for every change in subledger number.

2. The general ledger field is wired from columns 30 through 32 of both second and third reading to a comparing entry. The corresponding comparing exit is wired to program start intermediate to begin a second program cycle for any change in general ledger number.

3. Department is wired from columns 36 through 38 of both second and third reading to comparing entry. The comparing exit is wired to program start major to begin a third program cycle for every change in department number.

4. Each available counter in the machine has a pair of common total entry hubs (GG-HH, 51-56). When they are impulsed, the

counter will read out to print a total and reset by clearing to zero. Counter 8A is wired to read out and reset on a minor program, counter 8B on an intermediate program, and 8C on a major program. The counter exit and entry wiring is not shown in the diagram.

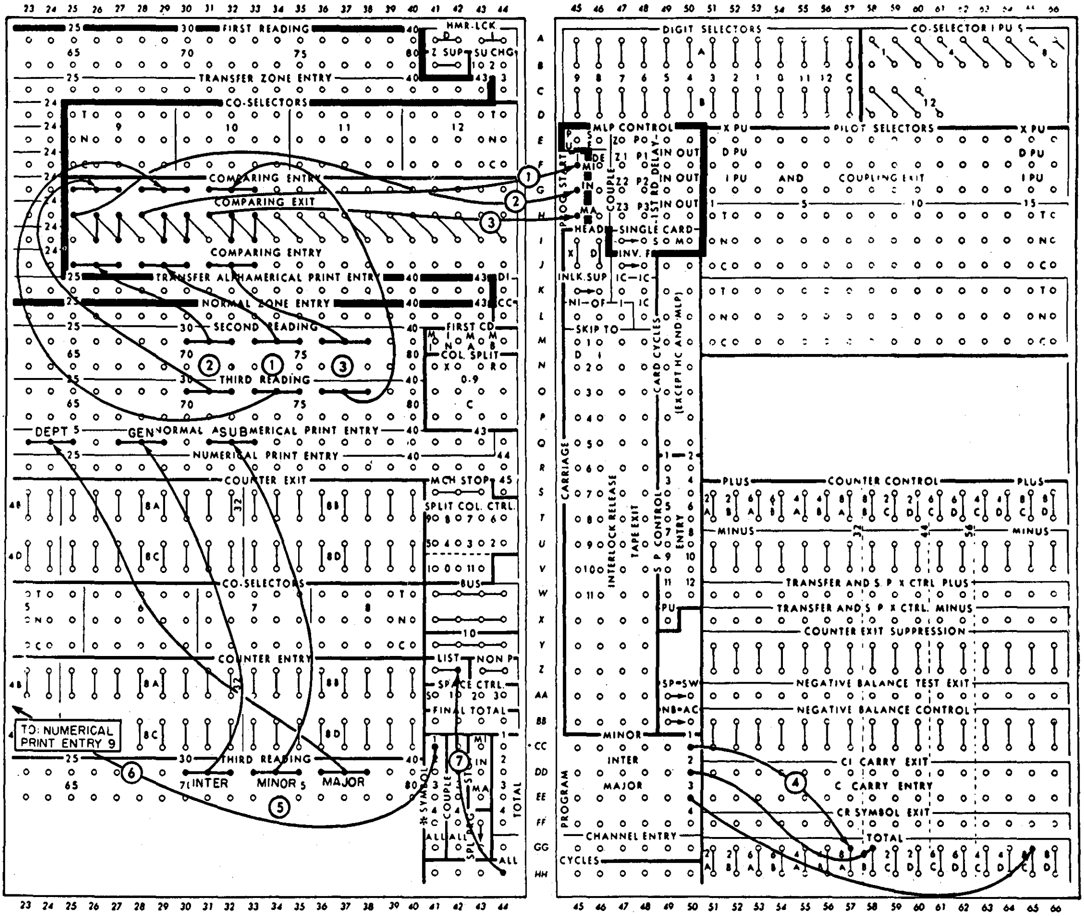


Figure 12.20 Numerical programming.

5. Subledger, general ledger, and department are listed for each card by wiring the three fields directly to alphamerical print entry.

6. An asterisk (*) is printed to the right of every minor total by wiring asterisk symbol hub 1 to numerical print entry 9. Asterisks may be printed from all odd-numbered numerical typebars.

7. The machine is wired to detail print every card.

Figure 12.21 shows that alphabetic as well as numerical information can be compared and printed. To do this, the second reading station is used in two ways: to read the zones for printing and to impulse one side of the comparing entry. Therefore, *split wires* are used because there is only one set of second reading hubs on the panel. As the name implies, a split wire is one with three jack plugs joined together by lengths of insulated wire. The use of split wires has the effect of reducing the total number of hubs needed on the control panel, with a corresponding reduction in panel size and complexity.

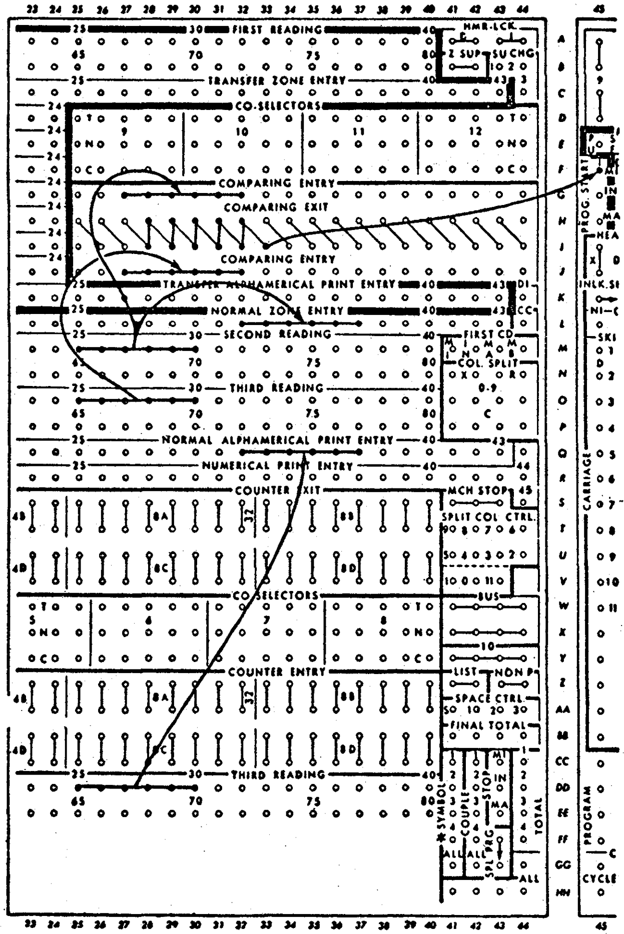


Figure 12.21 Alphabetic programming.

The Accounting Machine can be wired for program control for either detail or group printing. As previously explained, when the list hub is impulsed from all cycles, the typebars rise as each card passes through the machine. Consequently, all card information will be listed that is properly wired to print entry from the reading hubs. The speed of operation is 50 or 150 lines per minute, depending upon the model.

If the list hub is *not* impulsed, and program start is *not* wired, cards feed through the machine and totals can be accumulated, but no printing cycles occur. In this case, the machine is essentially an accumulating device or adding machine with final totals taken manually after all cards have been fed. This type of operation is often used to balance or establish control totals for an entire file or deck of cards.

If the list hub is not impulsed, but program start is wired from comparing entry, machine operation is as follows:

a. The typebars rise for the first card of each control group to print a *group indication*. One result of this kind of operation can be illustrated by referring again to the two reports in Figures 12.1 and 12.2. On the group printed report, note that only the control fields (department or branch, general ledger, standard ledger) are listed. These numbers are printed during the group indication cycle from the first card of the group. Other detail such as invoice number and date cannot be listed, except from the first card. After the group indication cycle, cards continue to feed without typebar operation until a change in control fields is sensed by the comparing unit. As cards continue to feed, or *tabulate*, totals are accumulated in the various counters as wired on the panel.

b. After the last card of a group passes third reading, card feeding stops while minor, intermediate, or major totals are printed. The counters involved are cleared and reset to zero. The form is spaced or ejected, the first card of the next group is indicated, and tabulating continues. Note that there is complete flexibility as to the number of cards that may be contained within any particular group. Thus, the number of lines shown on the report in Figure 12.2 corresponds to the number of control field changes in the file; any number of cards could be contained in each group.

Questions and Exercises

1. Explain the difference between detail printing and group printing.
2. What is summary punching?
3. What is the relationship between the punched control tape and the form shown in Figure 12.4?
4. Explain the difference between serial printing and parallel printing.
5. Design a report to show the detailed activity of withdrawals from the stock room. Use the requisition cards completed in exercise 8, chapter 11. It is assumed the cards are still in item number sequence. Each card is to be detail printed as follows:

Item number	
Quantity used –	minor total
Unit of measure	
Description	
Unit price	
Total cost –	minor total and final total
Department using	

6. Can this report be used to check the calculation of *Quantity* × *unit price* = *total cost* from the previous operation? How?
7. Assume the requisition cards are now sorted by department number. Design a report to show stationery expense for each department for a given period as follows:

Department	
Item number	
Quantity	
Unit measure	
Description	
Unit price	
Total cost –	minor total by item, intermediate total by department, final total

8. Can you demonstrate to the company auditors that this final report includes *all* items issued to each department for the period? Discuss this point in class.
9. Can additional controls be justified? If so, present your suggestions in class.
10. Can this report be used to prevent unauthorized use of stationery supplies? Also discuss in class.

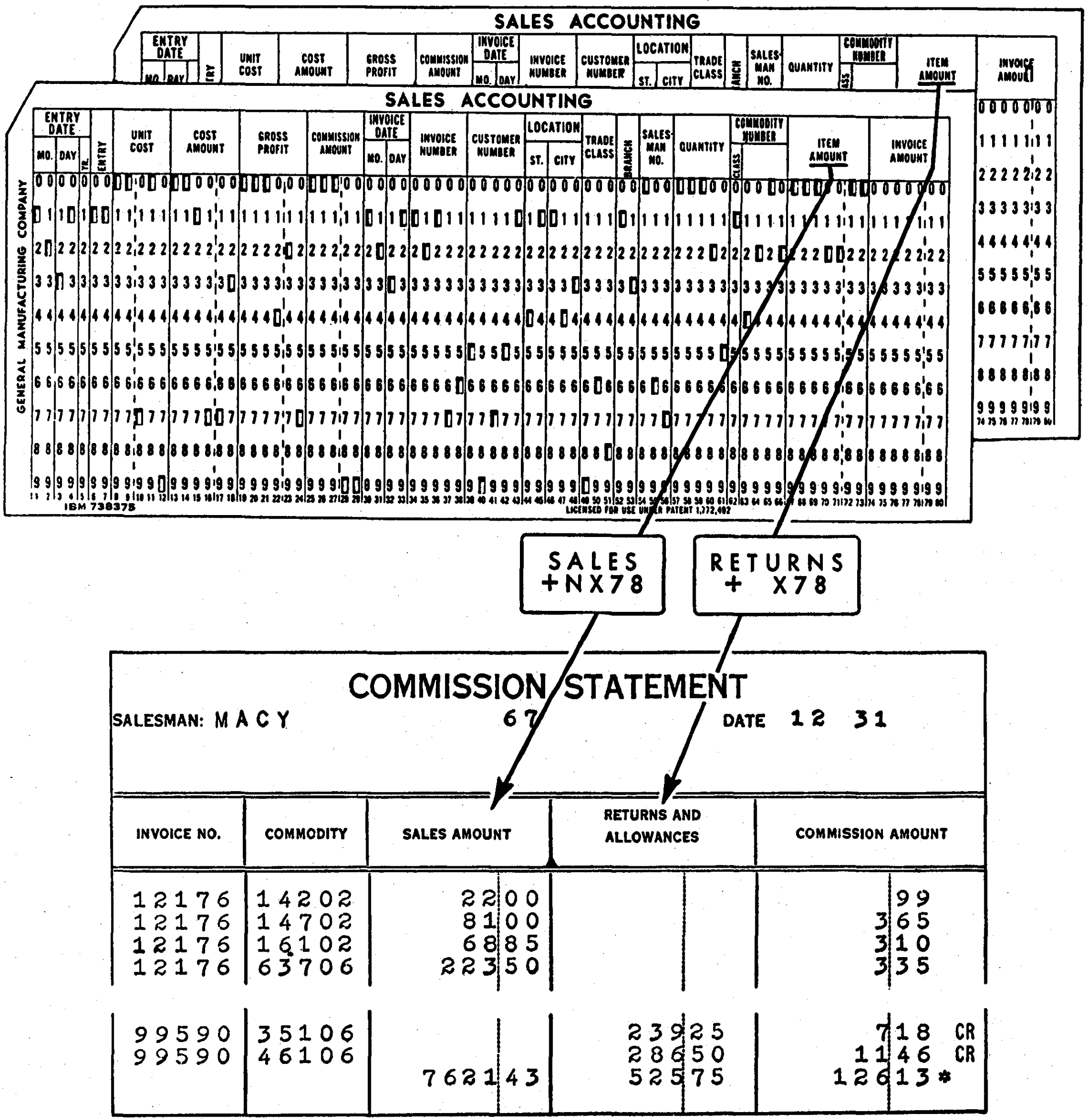
BASIC CONTROL PANEL WIRING 13

This chapter presents additional information about control panel wiring for IBM machines. Only elementary principles are discussed, using the 402 Accounting Machine for examples. The chapter illustrates typical machine capabilities that can be utilized in the design and implementation of business data processing systems. It is not intended to provide detailed training in wiring skills, although the examples presented should be understood in preparation for technical competence in this area.

If equipment is available for practice and demonstration, at least some of the examples can be machine run with the card forms shown and punched with sample data. The wiring diagrams can be used as a source of information for preparing or planning the stationery expense report designed as an exercise following chapter 12.

Counter Control

Figure 13.1 shows how the machine function of subtraction can be used for report printing. Negative fields in a sales accounting file are identified by a distinguishing X punch in column 78; positive fields are not punched with an X. A salesman commission statement is prepared from the file with invoice number, commodity number, sales amount, and commission amount detail printed. To accurately accumulate the total commission amount for each salesman, all commissions for current sales must be added. However,



COMMISSION STATEMENT

SALESMAN: M A C Y 67 DATE 12 31

INVOICE NO.	COMMODITY	SALES AMOUNT	RETURNS AND ALLOWANCES	COMMISSION AMOUNT
12176	14202	22 00		99
12176	14702	81 00		3 65
12176	16102	68 85		3 10
12176	63706	223 50		3 35
99590	35106		239 25	7 18 CR
99590	46106		286 50	11 46 CR
		7621 43	525 75	126 13 *

Figure 13.1 Subtraction for report printing.

amounts must be subtracted for commissions previously paid on goods which were later returned by customers.

Actual electromechanical operation of counter wheels to do subtraction need not be explained here, but the use of the *pilot selector* is a basic principle of machine control that is common to all punched-card equipment having control panels.

For example, the 402 is equipped with a number of pilot selectors, so-named for the guiding function they perform. Pilot selectors can be used independently or in conjunction with other selectors on the panel, called *coselectors.* Pilot selectors are located in rows E-M, holes 51-56, as shown in Figure 13.2.

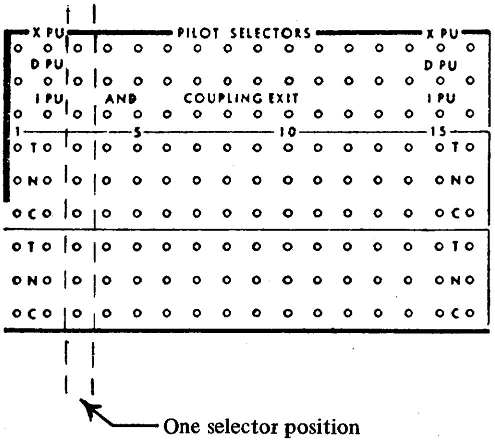


Figure 13.2 Pilot selectors.

The action of a selector is the same as that of a switch. Impulses from an exit hub on the control panel can be directed to either of two paths by opening or closing the switch with a pickup hub. There are two switches or *positions* for each selector arranged vertically on the panel.

The pickup can be actuated by either an X or digit impulse from a card column sensed at the second reading station. If X punches are to control the selector, second reading is wired to the X pickup. If a digit punch in the column is to control the selector, second reading is wired to the D pickup. The D pickup accepts any impulse from 9 through 12, while the X pickup accepts only X or 12 impulses. If a specific digit is to actuate the pickup, and all other possible punching in the column is to be ignored, then the digit must also be selected by another device in the machine before it is wired to the D pickup of the pilot selector.

An immediate pickup is also provided for each selector which will accept any impulse. This pickup transfers the selector immediately, instead of one cycle later. Also, whenever X or D hubs are impulsed, the 1 hub emits during the selector transfer for the *fol-*

lowing cycle. This impulse can be used to expand the pilot selector beyond two positions by picking up a coselector. When used in this manner, the I pickup is called a *coupling exit*. Thus, any coselector to which it is coupled functions exactly like the pilot selector.

If the column of the card with the distinguishing X punch is wired to the X pickup of a pilot selector, one machine cycle later a card cycles impulse wired to the C hub of the selector will be available at the T hub. Or, the card cycles impulse can be wired to the T hub where it will be available at C after the selector is transferred.

If there is no X punch in the card, one cycle later a card cycles impulse wired to the C hub of the selector will be available at the N side of the selector, or vice versa. Thus, if some cards of a file contain an X punch and others do not, certain functions of the machine can be correspondingly actuated or *not* actuated, whichever is desired. The most common use of the pilot selector is to control counters. A schematic of this operation is shown in Figure 13.3.

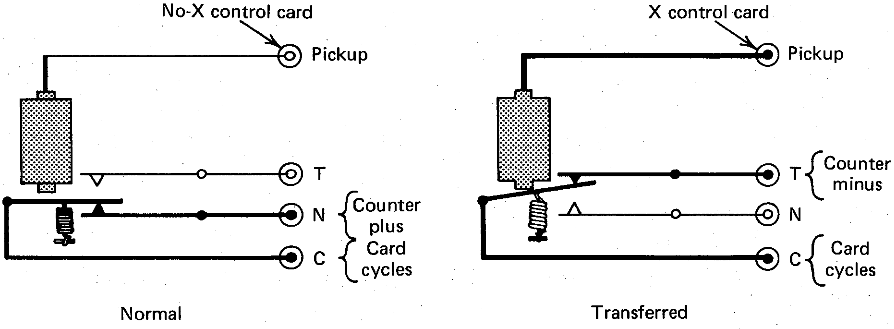


Figure 13.3 Schematic. Counter control with pilot selectors.

Assume that cards in a file to be added on a report are No-X and that cards to be subtracted are punched X in column 78. The X pickup of a selector is wired to second reading hub 78. Card cycles is wired to C. When there is no X in column 78, a card cycles impulse is directed to counter plus. When there is an X in column 78, card cycles is directed to counter minus. The counter consequently adds or subtracts as guided by the pilot selector.

Note that the counter can just as easily be controlled to add X cards and subtract No-X cards. Also, by using two separate counters, one counter can add No-X cards and the second counter

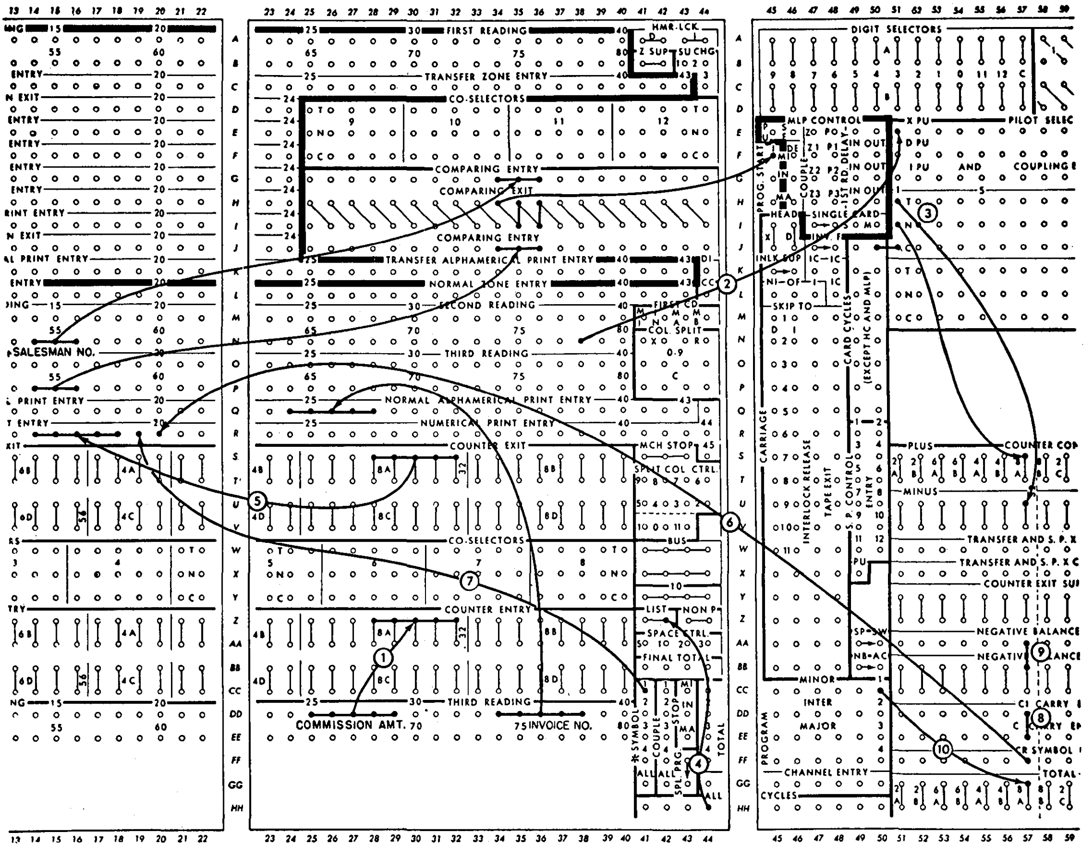


Figure 13.4 Wiring diagram. Subtraction and detail printing.

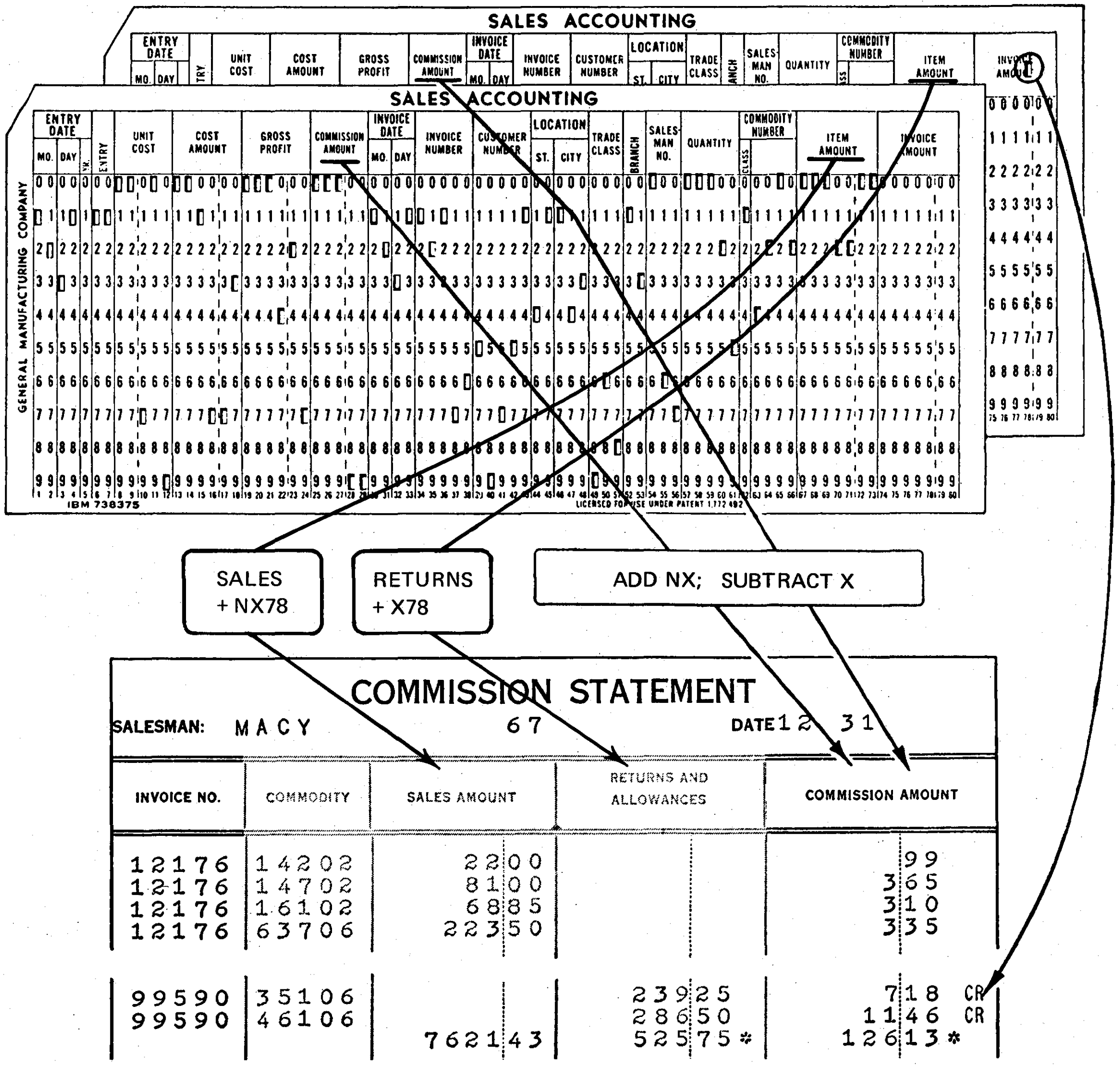
can add X cards. In this way, a separate total can be accumulated for commissions on returns or charge-backs as well as on sales.

Figure 13.4 shows the 402 control panel wiring for the report in Figure 13.5.

1. Commission amount in columns 25 through 29 is wired from third reading to counter 8A entry.

2. The X in column 78, identifying credit cards, is wired from second reading station to the X pickup of pilot selector 1. The X must be read from the second reading station so that the pilot selector will be transferred by the time the card reaches the third reading station.

3. A card cycles impulse is wired to the common of the pilot selector. An X in the card transfers the selector, and the card



COMMISSION STATEMENT

SALESMAN: MACY 67 DATE 12 31

INVOICE NO.	COMMODITY	SALES AMOUNT	RETURNS AND ALLOWANCES	COMMISSION AMOUNT
12176	14202	22 00		99
12176	14702	81 00		3 65
12176	16102	68 85		3 10
12176	63706	223 50		3 35
99590	35106		239 25	7 18 CR
99590	46106		286 50	11 46 CR
		7621 43	525 75 *	126 13 *

Figure 13.5 Commission statement.

cycles impulse controls counter 8A to subtract. A No-X card does not transfer the selector and the card cycles impulse controls counter 8A to add.

4. An all cycles impulse is wired to list to detail print every card in the file.

5. Both the detail amount and the total amount are obtained by wiring from 8A counter exit to numerical print entry 14-18.

6. Credit listings and credit totals are identified by wiring the credit symbol exit of 8A to numerical print entry 20.

7. Minor totals are identified by wiring the asterisk (*) symbol hub minor to print entry 19.

8. Whenever amounts are to be subtracted in a counter, the carry exit and carry entry hubs must be connected. These hubs have two functions: counter coupling and carry-back in subtraction. Two or more counters, up to a maximum of 16 positions, may be coupled together by wiring the CI of the counter containing the units position to the C of the coupled counter. A 12-position counter is obtained by coupling two 6-position counters, an 8-position and a 4-position counter, or three 4-position counters.

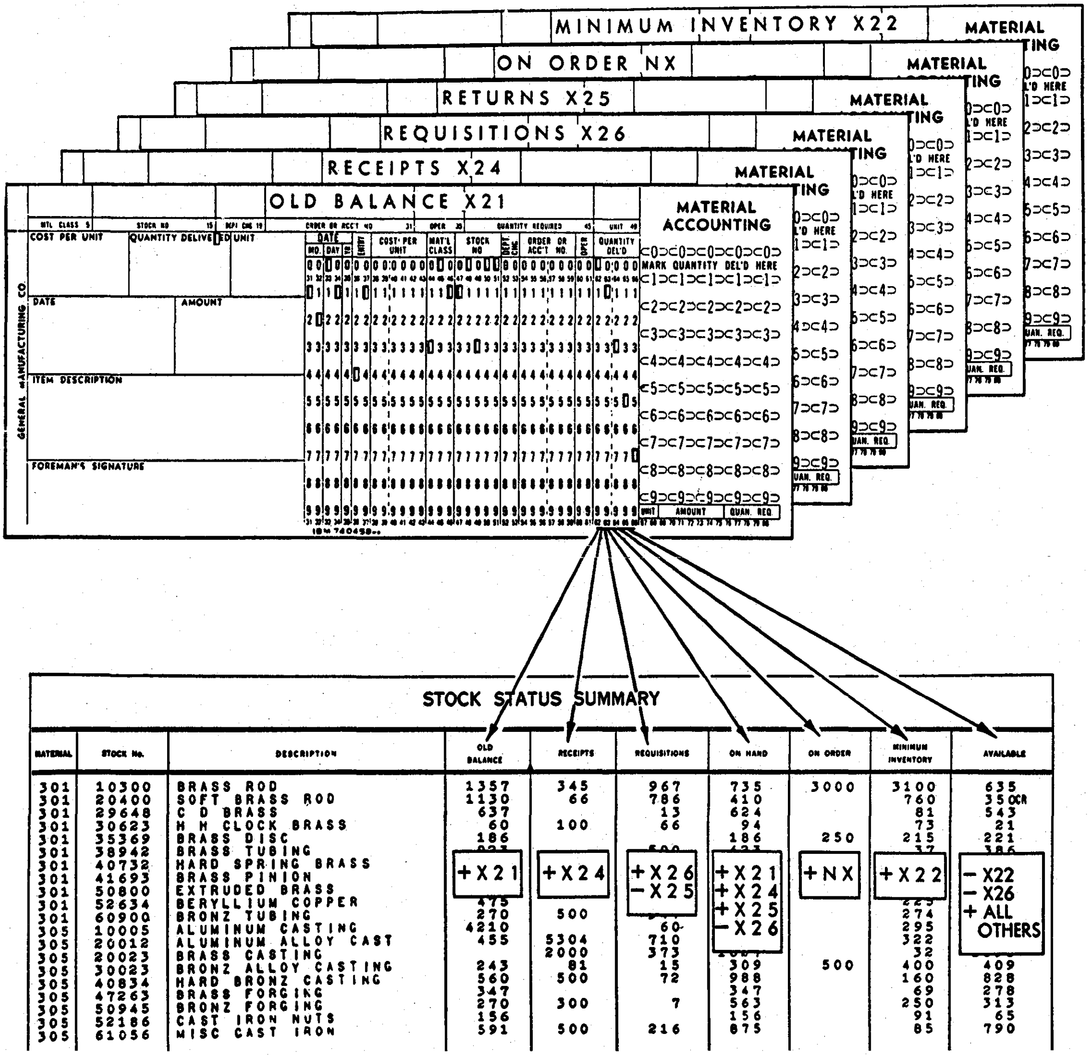
9. The counters in the 402 subtract by adding the 9s complement of the number. The complement amount standing in the counter is converted to a true figure by wiring the negative balance control of 8A. Negative balance test of 8A emits an impulse on total program whenever there is a 9 standing in the left or high-order position of 8A. Negative balance control receives an impulse from negative balance test exit on a total program to cause conversion factors to be added to the complement figure. The counter then prints the true balance as shown in the sample report.

10. Counter 8A is cleared for total printing on a minor total program. The program is initiated by an impulse from the comparing exit resulting from a change in salesman number.

Multiple Selection

In many punched card applications, a single card field often contains amounts or quantities for several different types of transaction. Each transaction must be distinguished from the others by an identifying punch. Figure 13.6 shows quantities for six different types of material accounting transaction punched in columns 62 through 66. Each transaction is identified by a significant X punch as follows:

Old balance	X 21
Receipts	X 24
Requisitions	X 26
Returns	X 25
Minimum inventory	X 22
On order	No-X



STOCK STATUS SUMMARY

MATERIAL	STOCK No.	DESCRIPTION	OLD BALANCE	RECEIPTS	REQUISITIONS	ON HAND	ON ORDER	MINIMUM INVENTORY	AVAILABLE
301	10300	BRASS ROD	1357	345	967	735	3000	3100	635
301	20400	SOFT BRASS ROD	1130	66	786	410		760	350CR
301	29648	C D BRASS	637		13	624		81	543
301	30623	H H CLOCK BRASS	60	100	66	94		73	21
301	35369	BRASS DISC	186			186	250	215	221
301	38942	BRASS TUBING	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	37	386
301	40732	HARD SPRING BRASS	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
301	41693	BRASS PINION	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
301	50800	EXTRUDED BRASS	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
301	52634	BERYLLIUM COPPER	475	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
301	60900	BRONZ TUBING	270	500	[illegible]	[illegible]		274	[illegible]
305	10005	ALUMINUM CASTING	4210		60	[illegible]		295	[illegible]
305	20012	ALUMINUM ALLOY CAST	455	5304	710	[illegible]		322	[illegible]
305	20023	BRASS CASTING		2000	373	[illegible]		32	[illegible]
305	30023	BRONZ ALLOY CASTING	243	81	15	309	500	400	409
305	40834	HARD BRONZ CASTING	560	500	72	988		160	828
305	47263	BRASS FORGING	347			347		69	278
305	50945	BRONZ FORGING	270	300	7	563		250	313
305	52186	CAST IRON NUTS	156			156		91	65
305	61056	MISC CAST IRON	591	500	216	875		85	790

Figure 13.6 Stock status summary and material accounting file.

These distinguishing punches are used to control seven counters to add or subtract as required to print a stock status summary, also shown in Figure 13.6.

The wiring diagram is shown in Figure 13.7.

1. One pilot selector is picked up on each of the above columns that may be punched with an X.

2. A card cycles impulse is wired through the selectors as follows:

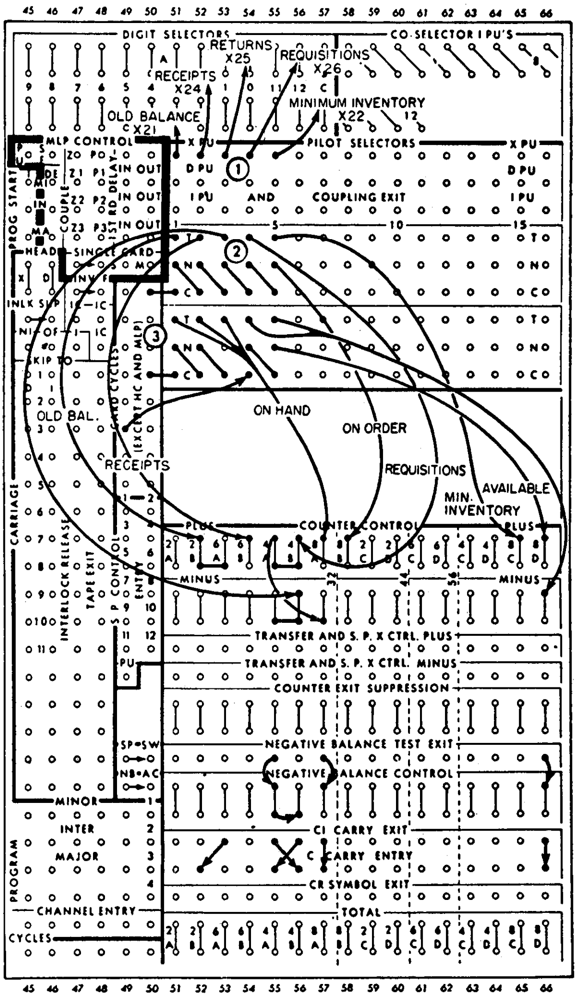


Figure 13.7 Wiring diagram, multiple X selection.

a. Through the transferred side of pilot selector 1 to add old balance in counters 2A and 6A when there is an X punched in column 21. Counters 2A and 6A are coupled.

b. Through the transferred side of pilot selector 2 to add receipts in counter 6B when there is an X punched in column 24.

c. Through the transferred side of pilot selector 3 to subtract returns in 4A and 4B when there is an X punched in column 25. Counters 4A and 4B are coupled.

d. Through the transferred side of pilot selector 4 to add requisitions in 4A and 4B, and to subtract requisitions in the on-hand counter 8A. Requisitions are punched X in column 26.

e. Through the transferred side of pilot selector 5 to add minimum inventory in 8C.

f. A card cycles impulse is wired through the normal hubs of each selector and is available out of the normal side of selector 5 if a card does not contain an X punch. It is used to add on order cards in 8B.

3. A card cycles impulse is wired through the transferred side of the first three pilot selectors to add old balance, receipts, and returns in the on-hand counter 8A. A card cycles impulse is wired through the transferred side of the fourth and fifth selectors to subtract requisitions and minimum inventory in the available counter 8D. All other cards add in 8D.

Digit Selection

Various types of cards in a file can also be identified with digits punched in a single column. This method is usually more convenient, because transaction cards can be quickly selected or separated by type in one pass of the file through a sorter.

For example, assume that column 30 of the material accounting cards shown in Figure 13.6 is reserved for punching type of transaction, coded as follows:

Old balance	Digit 1
Receipts	Digit 2
Requisitions	Digit 3
Returns	Digit 4
Minimum inventory	Digit 5
On order	No digit

Figure 13.8 shows the change in wiring required to handle selection by digits. Column 30 is wired from second reading to the C hub of a *digit selector*.

There are two digit selectors on the control panel, A and B, located at A-D, 45-57. Each selector has a pair of common (C) entry hubs and a pair of exit hubs for each of the digits 9 through

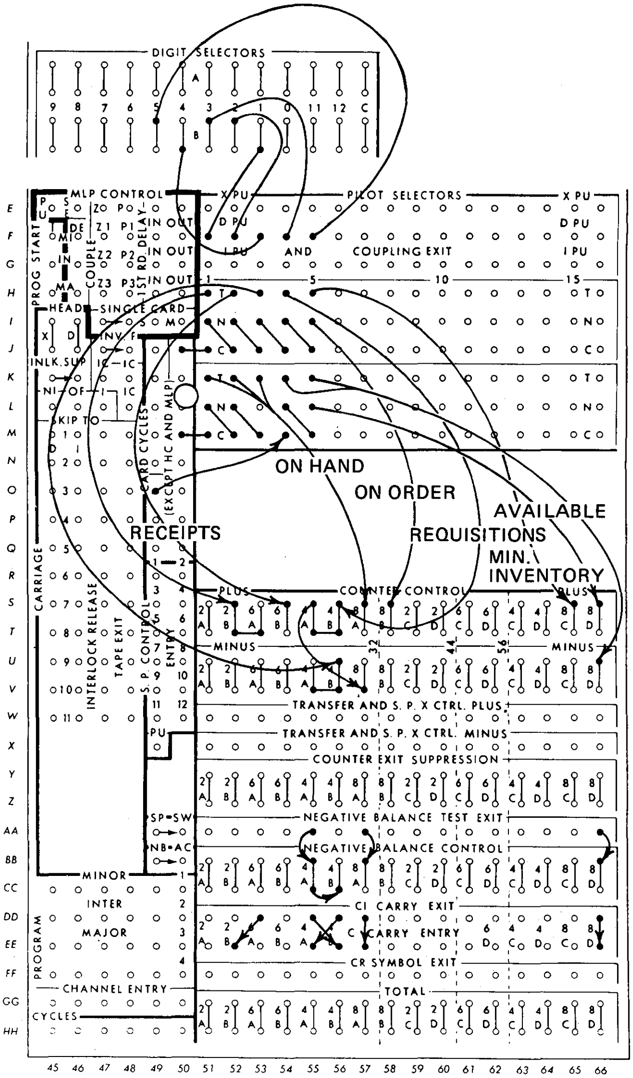


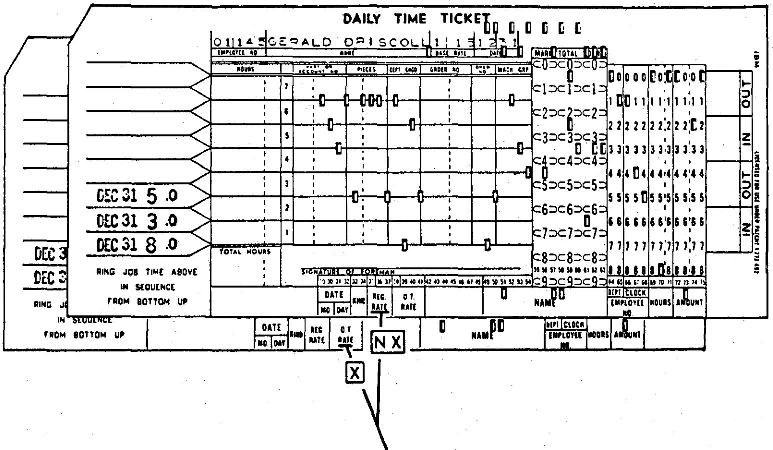
Figure 13.8 Digit selection.

0, and for 11 and 12. When C is wired from a card column, the hubs from 9 through 12 will emit impulses corresponding to the digits punched in the column. These digits may be used to pick up pilot selectors or to operate functions of the machine that may be digit controlled.

Digit selectors are operative on card feed cycles only. In Figure 13.8 one pilot selector is picked up on each of the above digits.

Field Selection Different fields from different types of cards can be printed in the same typebars or accumulated in the same counter of the accounting machine. This operation is called *field selection*. It is done by the use of coselectors controlled by an identifying punch.

Coselectors are so-named because they often operate in conjunction with pilot selectors. A coselector has five positions each with a common (C), normal (N), and a transferred (T) hub. In principle they function like pilot selectors. That is, when transferred, there is a connection between C and T; when not transferred, there is a connection between C and N. Coselectors are located at W-Y, 1-40; on the control panel.



LABOR DISTRIBUTION REGISTER

EMPLOYEE NAME	EMPL. No.		DATE		ENTRY	RATE	PART No.	OPER.	ORDER No.	HOURS	LABOR COST
	DEPT.	CLOCK	NO.	DAY							
GERALD DRISCOLL	1	145	12	31	51	115				80	920
	1	145	12	31	51	115	11872	2	109396	20	230
	1	145	12	31	51	115	11892	2	309397	60	690
	1	145	12	31	52	1725				20	345
	1	145	12	31	51	1725	11872	2	109396	20	345
										100 *	1265 *
										100 *	1265 *
JAMES DUHLMEIER	1	150	12	31	51	65				80	520
	1	150	12	31	51	65	12067	3	409399	80	520
	1	150	12	31	52	975				20	195

Figure 13.9 Labor distribution register and time tickets.

Each coselector has two common pickup hubs diagonally arranged for convenience in wiring. When these hubs are impulsed, the coselector transfers immediately and holds for the duration of the machine cycle. When these hubs are wired from the coupling exit (immediate pickup row) of a pilot selector, the coselector transfers with the pilot selector and holds transferred for the same length of time as the pilot selector. Therefore, when a coselector is picked up from the coupling exit of a pilot selector, the number of positions for selection is increased from 2 to 7.

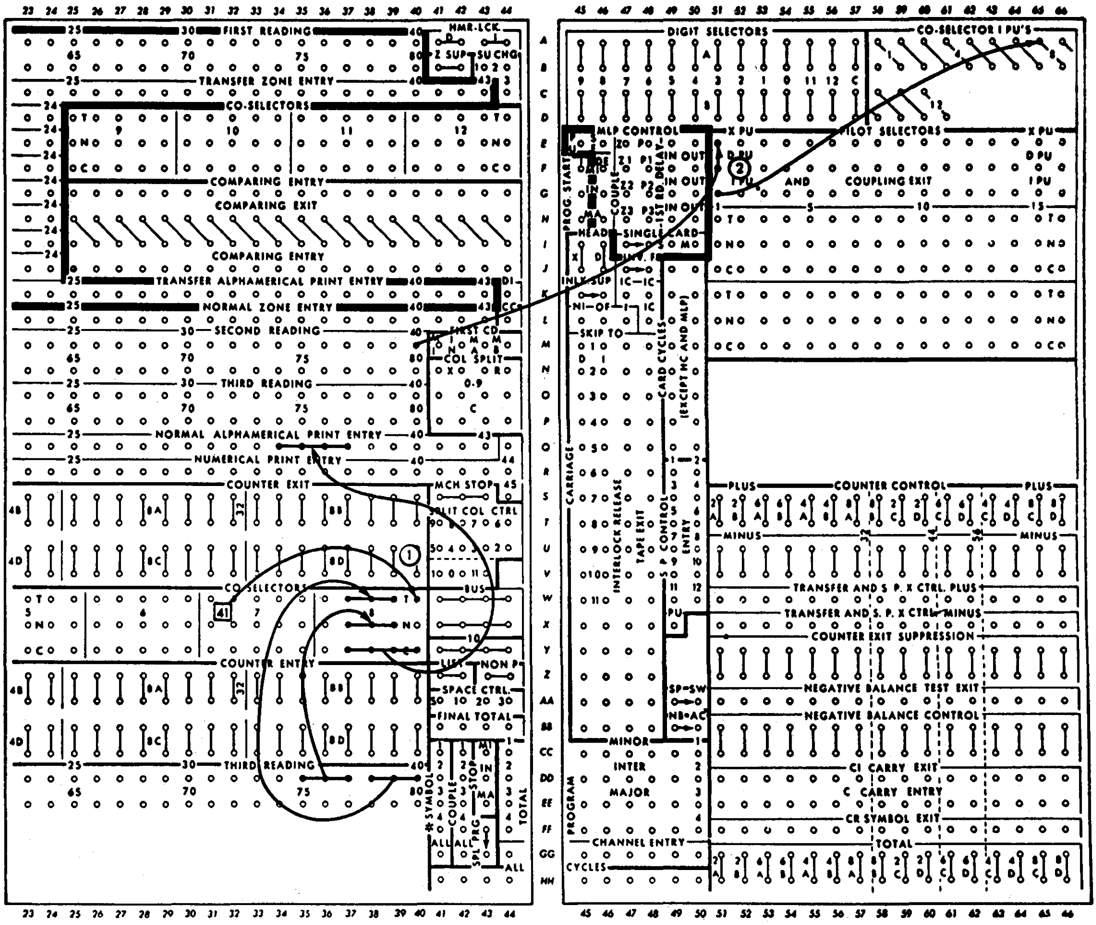


Figure 13.10 Wiring diagram, field selection.

Figure 13.9 shows a labor distribution register detail printed from a file of daily time tickets. Wages for employees are calculated from two different rates: regular and overtime. Time tickets for overtime are punched with an X in column 40; regular time tickets are No-X. Regular rate is punched in columns 35 through 37; overtime rate is punched in columns 38 through 41.

The rate of pay, either regular or overtime, is printed in only one column of the report, in alphanumerical typebars 34 through 37. The wiring for field selection is shown in Figure 13.10.

1. Regular rate is wired from third reading columns 35 through 37 to the normal side of coselector 8. Overtime rate is wired from columns 38 through 41 of third reading brushes to the transferred side of the same coselector. The common side is wired to alphanumerical print entry 34–37.

2. The coselector is picked up as follows. The controlling X in column 40 is wired from second reading to the X pickup of pilot selector 1. The coupling exit of the pilot selector is wired to the pickup of coselector 8. The coselector transfers exactly like the pilot selector. When the No-X card passes the third reading, both selectors will be normal and the regular rate then prints from card columns 35 through 37. When the X card passes third reading, both selectors will be transferred and the overtime rate prints from card columns 38 through 41.

Either alphabetic or numerical information can be field selected. For this operation, one pilot selector and two coselectors are required. One coselector selects zoning and the other selects digits of the alphabetic field.

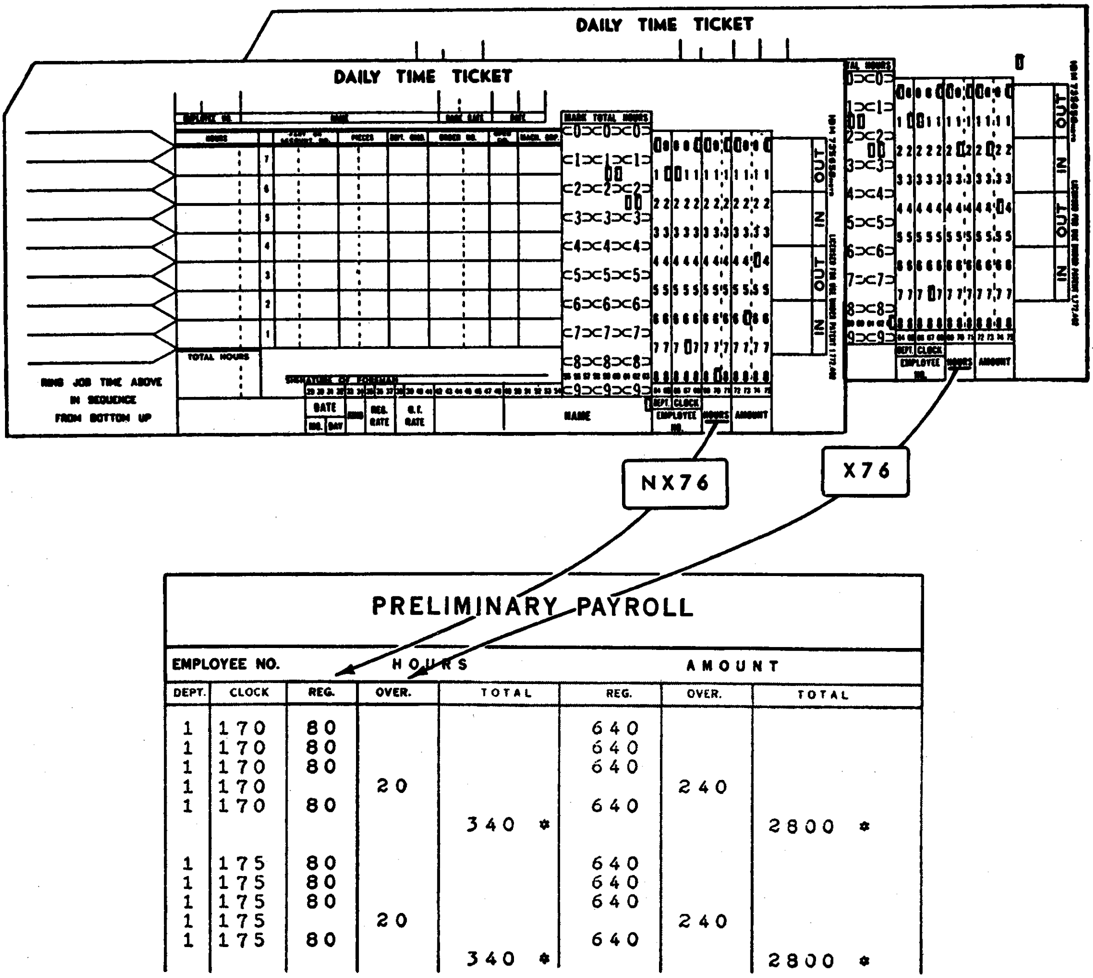
The selector used for zoning must be picked from second reading and in time for the 0, 11, or 12 zone punches. Therefore, if alphabetic field selection is to be used, the "X" card must also be punched with an identifying digit (1 through 9) to actuate the coselector. The coselector used to select the digit portion of the field must be controlled from the coupling exit of a pilot selector.

Class Selection

The same field from different types of cards can be printed in different printing locations, provided each type of card is identified with a distinguishing X or digit punch. Figure 13.11 illustrates this type of operation.

A preliminary payroll report is printed from daily time tickets. The tickets can show either regular or overtime hours, depending upon the employees' assigned working schedules. All overtime tickets are punched with an X in column 76.

When a card field is to be selected to one of two possible printing positions, the card field is wired from third reading to the common hubs of a coselector. The normal side of the selector is wired to the typebars where the No-X field is to print. The transferred side is wired to the typebars where the X field is to print.



PRELIMINARY PAYROLL

EMPLOYEE NO.		HOURS			AMOUNT		
DEPT.	CLOCK	REG.	OVER.	TOTAL	REG.	OVER.	TOTAL
1	170	80			640		
1	170	80			640		
1	170	80			640		
1	170		20			240	
1	170	80			640		
				340 *			2800 *
1	175	80			640		
1	175	80			640		
1	175	80			640		
1	175		20			240	
1	175	80			640		
				340 *			2800 *

Figure 13.11 Preliminary payroll report and time tickets.

Because the regular hours field in this example is punched in a No-X card, regular hours will be printed in alphamerical typebars 34 through 36, the regular hours column of the report. Hours in the X-76 card are overtime hours and will print in alphamerical typebars 38 through 40. Figure 13.12 is the wiring diagram for numerical class selection.

Alphabetic class selection is also possible with the use of two coselectors, one to select the numerical portion of the field and one to select the zone portion. However, the zone selector must be picked up from a 1-9 digit, instead of a single X in the control column.

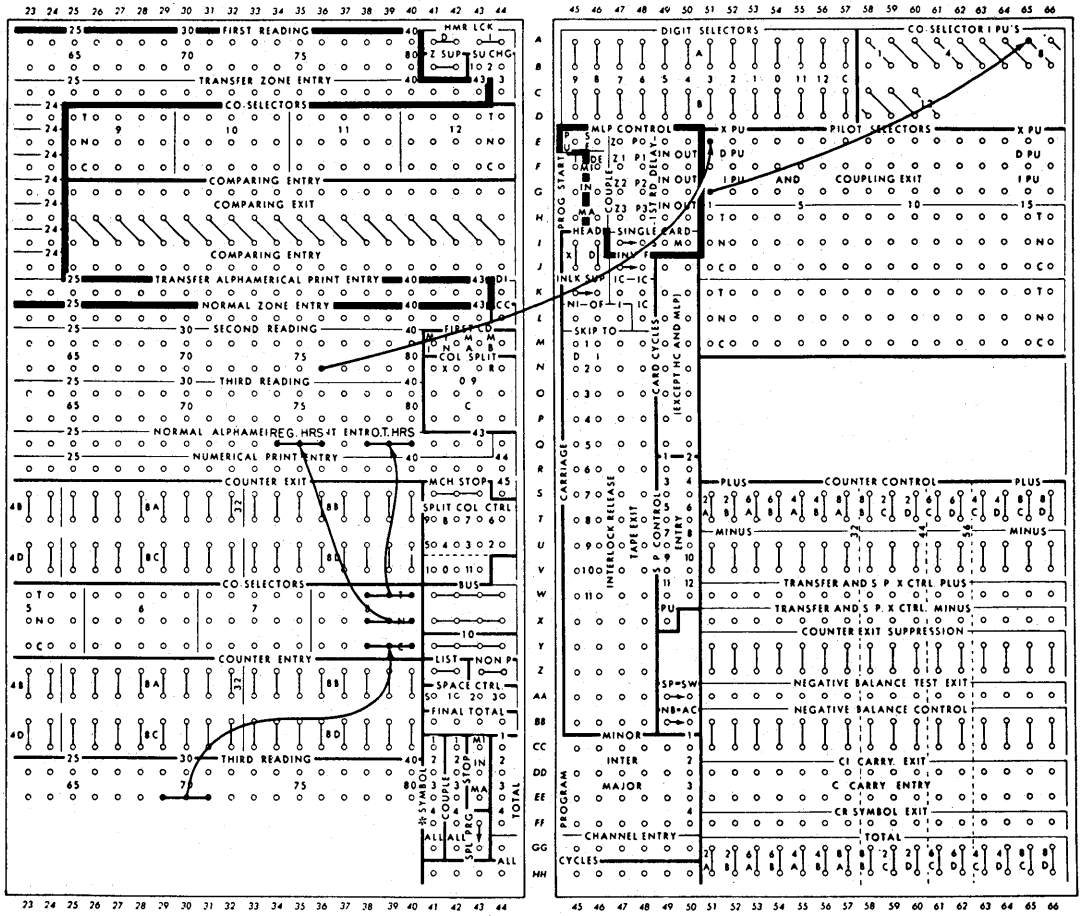


Figure 13.12 Wiring diagram. Class selection.

Additional Accounting Machine Features

A number of additional accounting machine features are available to increase the flexibility of report printing. The most important of these are summarized briefly in the following sections.

Crossfooting

As many as three card fields can be crossfooted on a standard 402. The fields can be read into counters and crossfooted for individual cards, or totals of the fields can be accumulated from a group of cards and then crossfooted into one counter. The use of the feature may save one pass of a file through the calculating punch since no field need be punched in the cards for the crossfooted total. The total only appears on the printed report. The three fields are entered into three counters as the card passes the third reading station. The fields may be added or subtracted to obtain the total, for example $A + B - C = T$.

Each field is accumulated in a separate counter of appropriate capacity, like 8A, 8B, or 8C. If the report is detail printed, the fields are also wired to normal print entry from third reading. All three counters are connected by wiring their exits together to provide a path for the transfer of totals from one counter to another during what would normally be a print cycle.

After each card, or at the end of a control group, the machine is programmed to transfer the minor total into the intermediate counter and then from the intermediate counter into the major counter. Printing can be suppressed during the minor and intermediate cycles, if desired.

Progressive Total Printing

Progressive totals can be printed by not resetting the counters during a print cycle. After printing, any total standing in the counter is retained and additional amounts can be accumulated as wired from the reading brushes.

On the 402, counter reset can be prevented by impulsing the plus and minus hubs of a counter control at the same time that the total hub is impulsed. This wiring allows the counter wheels to rotate past zero and to continue rotating to their original position at the beginning of the print cycle. For example, if a counter contains a digit 6 and is impulsed to clear normally, the counter wheel turns from 6 to 7 to 8 and on to zero where it stops. However, if the plus and minus hubs are both impulsed, the stop at zero is suspended and the wheel keeps turning until it reaches 6 again.

Although this feature is designed primarily for printing totals without clearing the counter, it may also be used for printing any constant numerical information, such as a date or identification number. Such information can in effect be stored in the counter from a master card at the beginning of a run and later read out as desired. For example, the date or identification number might be printed at the top or bottom of each new page of a report. Progressive totals can also be used to consecutively number report pages.

Special Program

On the standard 402 there are three types of program starts, resulting in a maximum of three total program cycles. When the minor program is impulsed, only one total program cycle is taken. When the intermediate program start is impulsed, two program cycles are taken, and when major program start is impulsed, three program cycles are taken.

When the machine is equipped with a special program device, any number of total program cycles may be initiated for any one program start. For example, four program cycles may be started by a minor start for the purpose of crossfooting four minor totals or printing them one underneath the other. As many as five or 10 totals can be crossfooted.

Summary Punching

Summary punching is the automatic preparation of one total card to replace an entire group of cards. The summary card normally contains the identification of the single group and one or more accumulated totals for that same group.

The preparation of summary cards reduces card volume for batch processing, particularly when the ratio of detail cards to control groups is large. When summarized, any given card file will be reduced in volume by the average number of details per group. That is, if there are an average of 10 details per group, the file will be reduced 10 to one. This reduction in card volume produces corresponding savings in machine processing time thereby also reducing the time between the end of an accounting period and the production of a finished report.

The preparation of summary cards also is an efficient method of carrying balances of quantities or amounts forward from one accounting period to the next. Stock status, year-to-date earnings, and accounts receivable are common examples of this practice, i.e., *Old balance ± current transactions = new balance*. Previous summary cards from last period's run are always punched with the *old balance*. *Current transactions* are accumulated and their effect on the *old balance* is summarized into a balance forward or *new balance* summary card. A report is printed as a record and to provide controls over the accuracy of the entire process.

Summary cards are punched by the attachment of a *summary punch* to the accounting machine by a cable. A number of units are attachable for this purpose, including the 519 Document-Originating Machine described earlier. When the control panels of both machines are properly wired, the exits of all the counters are available on the summary punch control panel. Data can be summarized into any fields of the summary card, although at least the control fields are normally punched in the same location from detail to new summary.

Note that only information *standing in counters* of the accounting machine can be summary punched. On a standard machine, this means only numerical information since counters are used to accumulate quantities or amounts. However, machines can be fitted with an alphabetic summary punching device for the transfer of group indicated alphabetic data.

Card feeding on the accounting machine is interrupted while a card is being fed and punched by the summary punch. Provision is also made for the transfer of credit balances to the summary card with indicating X punches in selected card columns.

Figure 13.13 is a sample of wiring that shows the relationship between an accounting machine and a summary punch. The accounting machine wiring has been previously shown for report printing (Figure 13.5) to produce a salesman commission statement. The following explanation describes the additional wiring required to summarize by salesman number and commission amount.

1. The summary punch switch is turned on. These two hubs on the accounting machine control panel must be connected to synchronize the operation of both machines. The hubs may also be connected through a pilot selector or a coselector to provide the ability to summary punch only selected groups of cards.

2. A summary card can be punched on any program change by impulsing the pickup hub of *summary punch control.* When a total program exit is wired to the pickup, and the summary punch switch is on, the summary punch will feed and punch just before the accounting machine prints the corresponding total. More than one class of total can be punched in the same run, e.g., minor and major. In this case, summary punch control is picked up from the minor program start. Therefore, a summary card will be punched for each salesman, because the minor program start exits an impulse every time a group of cards changes from one salesman number to another.

3. To identify credit balances in commission amounts, it is necessary to punch an identifying X in column 80 of the summary card. To satisfy this requirement, each counter in the accounting machine has a corresponding summary punch X control plus and X control minus. These hubs are on the accounting machine panel

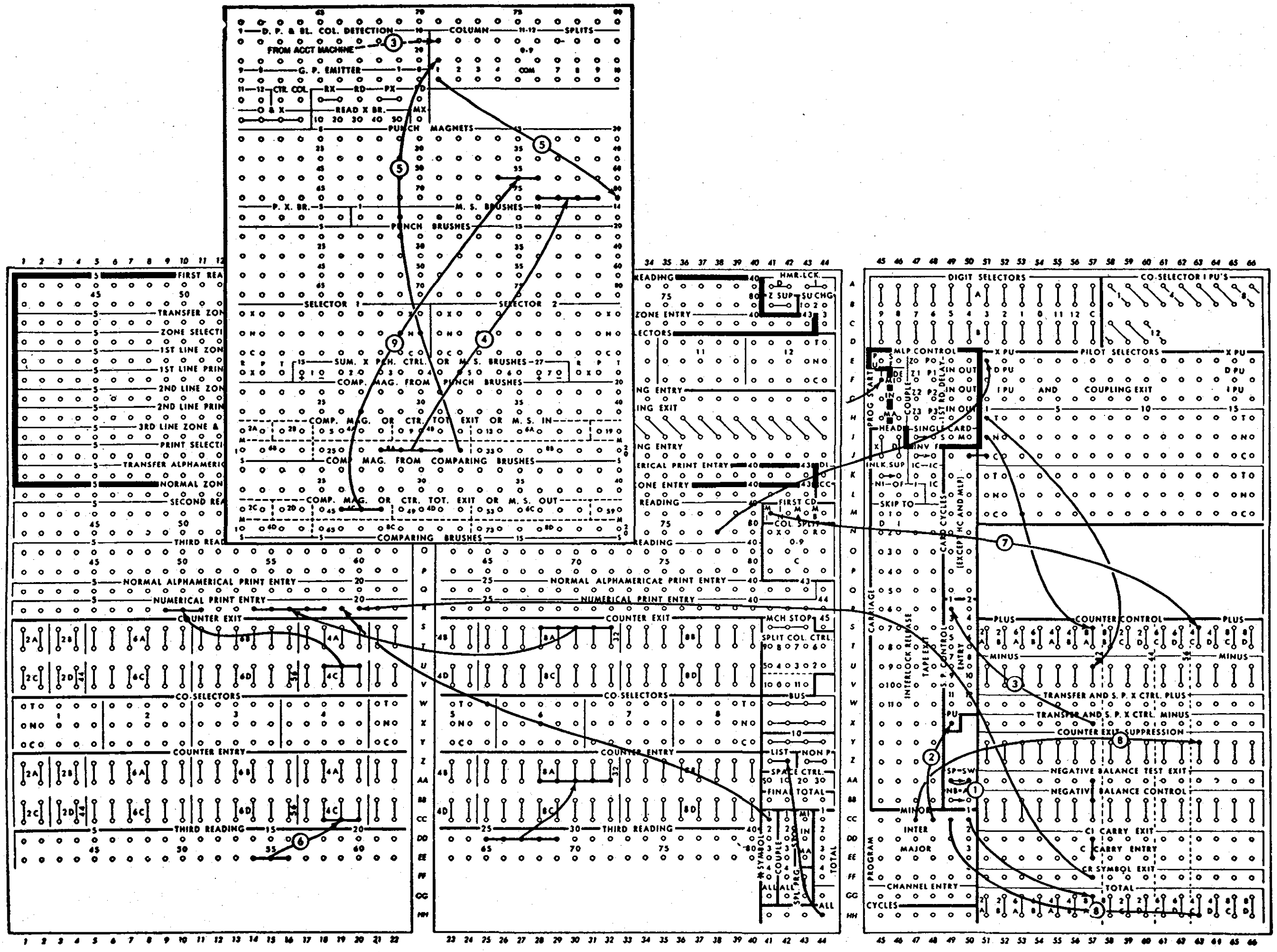


Figure 13.13 Wiring diagram. Net balance summary punching.

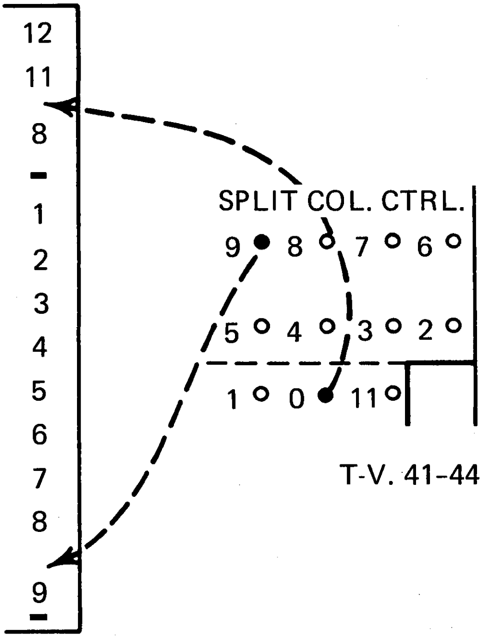


Figure 13.14 Summary punch X control.

in rows W, X, 51-66. Thus, an amount or quantity field can be punched with either a plus or minus X, whichever is required.

In the example, the summary punch X control minus of 8A is wired to summary punch entry 1. When the amount balance in counter 8A is minus, an impulse is transferred by internal wiring to the corresponding 11-12 column split hub on the summary punch, as shown in Figure 13.14. From column split 1 the impulse is transferred through the common hub to the punch magnets of the summary punch.

4. Counter exits are provided on the summary punch control panel for each counter in the accounting machine. To punch commission amount in columns 76 through 80 of the summary card, counter 8A exits are wired to punch, except the units position.

5. The units position of 8A is wired to column 80 through the 0-9 position of column split 1. A credit balance summary card will be identified by an X punch in column 80.

6. Salesman number is wired to counter 4C.

7. The salesman number will be used to identify the summary card, and hence must add in the counter only once for each salesman. Minor first card is wired to 4C plus, causing the first card of every group to add.

8. Counter 4C is cleared on every minor program. Printing of the group indication is suppressed in the total cycle.

9. Counter 4C exits on the summary punch are wired to punch columns 54 through 56.

Multiple Line Printing

It is often a decided advantage while handling many name and address files to be able to print at least three lines from a single card. Then, one card can contain all the elements of a normal address: name, street and number, city, state, and zip code as shown in Figure 13.15. The IBM 403 Accounting Machine is designed for this purpose with a multiple line printing (MLP) feature.

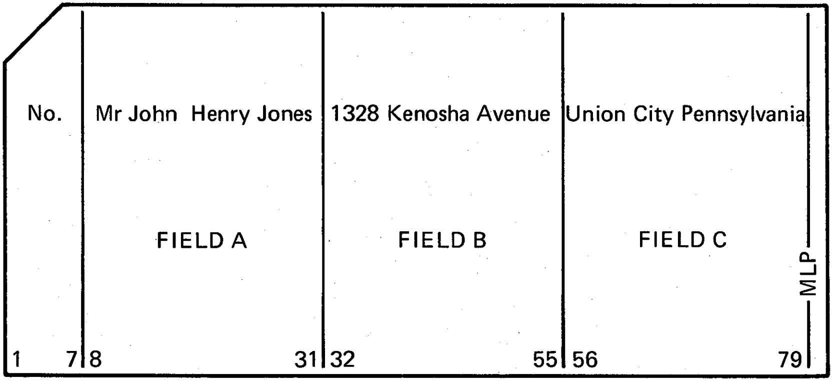


Figure 13.15 MLP three-card format.

The special MLP card is identified with a 9, an 8, and a 1, 2, or 3 punch in a designated card column. The 1, 2, or 3 punches indicate whether 1, 2, or 3 lines are to be printed from the card. Normally, 24 card columns are sufficient for each line of an address, but these may be expanded within the 80-column capacity of the card. One line may be made up of more than 24 columns; another line may be made up of less.

The 403 performs all the operations of the 402 and in the same manner. In addition, a control panel wired for the 402 can be used on the 403 without changing the wiring. To accomplish MLP, the 403 is equipped with a third set of 80 reading brushes located at a first reading station. Two other reading stations are located in the same position as on the 402. A fourth nonreading station is used for printing the third line of the three-line card. The arrangement of the four stations in the machine is shown in Figure 13.16.

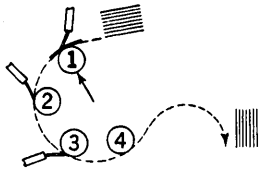
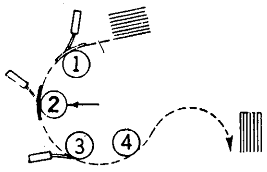
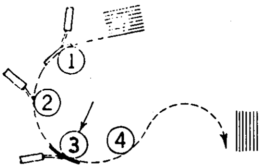
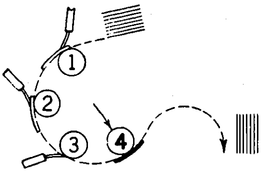
PATH OF CARD	PRINTING FROM CARD	OPERATION
		Zone punching from Field A zones the type bars.
	MR JOHN HENRY JONES	Digit punching in Field A is combined with zone punching for Field A, and Field A is printed. Zone punching from Field B zones the type bars.
	MR JOHN HENRY JONES 1328 KENOSHA AVENUE	Digit punching in Field B is combined with zone punching for Field B, and Field B is printed. The normal card following an MLP3 card remains stationary at the first station (between 0 and 1) as the second line prints. Digit punching for Field C is stored in counters. Type bars are zoned for Field C.
	MR JOHN HENRY JONES 1328 KENOSHA AVENUE UNION CITY PENNSYLVANIA	Digit punching and zone punching for Field C are combined, and Field C is printed. The normal card advances from the first to the second station.

Figure 13.16 Schematic, MLP operation.

1. Field A, First Line. The zone punches of field A are wired on the control panel to read at station 1 and the designated typebars are zoned. The card may be held at this first station between the 1 and 0 positions before reading the zone punches. This delay is necessary to prevent overlap in zoning because the same typebars cannot be zoned simultaneously from two different cards – one at the first station and one at the second. The delay in feeding

may vary from one to two cycles depending upon the type of card being read and the relationship of one card to another.

Digit punches are read at station two and field A prints.

2. Field B, Second Line. Zone punches for field B are read at station 2 and typebars are zoned. Digits punches for that field are read at station 3 and field B prints. Note that the second line is printed in the same typebar positions as the first line.

3. Field C, Third Line. Zone punches for field C are read at station 3 and the typebars are zoned. Digit punches are also read at station 3 and are stored in 24 counter positions set aside for this purpose: counters 2A,B, 6A,B, and 4A,B. The digit punching is combined with the zoning at station 4 when the counters are read out and field C is printed. Field C also prints in the same typebar positions as fields A and B.

The use of the 24 counter positions for storage of third line digits prohibits the use of these counters for totals that are not cleared before the next MLP card is printed. That is, these counters cannot overlap the storage of digits for line printing with total accumulation.

Normal cards may be interspersed with MLP cards, thereby providing for the use of the feature with such common data processing applications as invoice writing, billing, and the like. The MLP feature may also be used for crossfooting three fields.

Questions and Exercises

1. Explain the operation of the pilot selector and coselector in relation to the timing of cards as they pass through the accounting machine.
2. What is the purpose of counter coupling?
3. How does multiple selection expand the data recording capacity of punched cards?
4. Why is digit identification of cards in a file often more convenient than X identification?
5. How does field selection differ from pilot selection and coselection?
6. What is the difference between field selection and class selection?
7. What is the purpose of the multiple line printing feature?
8. Prepare a 402 Accounting Machine wiring diagram to produce the stationery expense report designed as an exercise following chapter 12.

IBM 407 ACCOUNTING MACHINE 14

The IBM 407 Accounting Machine also prepares printed reports from 80-column punched cards. Several advanced features give this machine considerably more flexibility, speed, and capacity than the earlier 402 and 403 models. Most important is its unique method of reading cards. Figure 14.1 is a schematic of the 407 feed unit.

Cards are placed in the hopper with the 9-edge toward the throat. Cards feed into the machine from the bottom of the deck in the same sequence as the printed report. Each card is positioned at the first, then at the second reading station with card grippers that move horizontally as indicated by the arrows in the diagram. Cards can be held at the reading stations for any given number of cycles. After reading, they move around a rotating drum into the stacker and are held there by a pressure plate. When the stacker is full, the machine stops.

As a card is positioned at the reading station, it is lined up by the card aligners so that the 960 possible punching positions are directly under 960 stationary reading brushes. The brushes are directly above 960 metal segments, labeled 9, 8, 7, ..., 12 in Figure 14.1. Any hole punched in a card allows the corresponding brush to make contact with a metal segment. The resulting electrical impulse is transmitted from a commutator as it rotates clockwise in relation to the brushes. There are 80 commutators at each read-

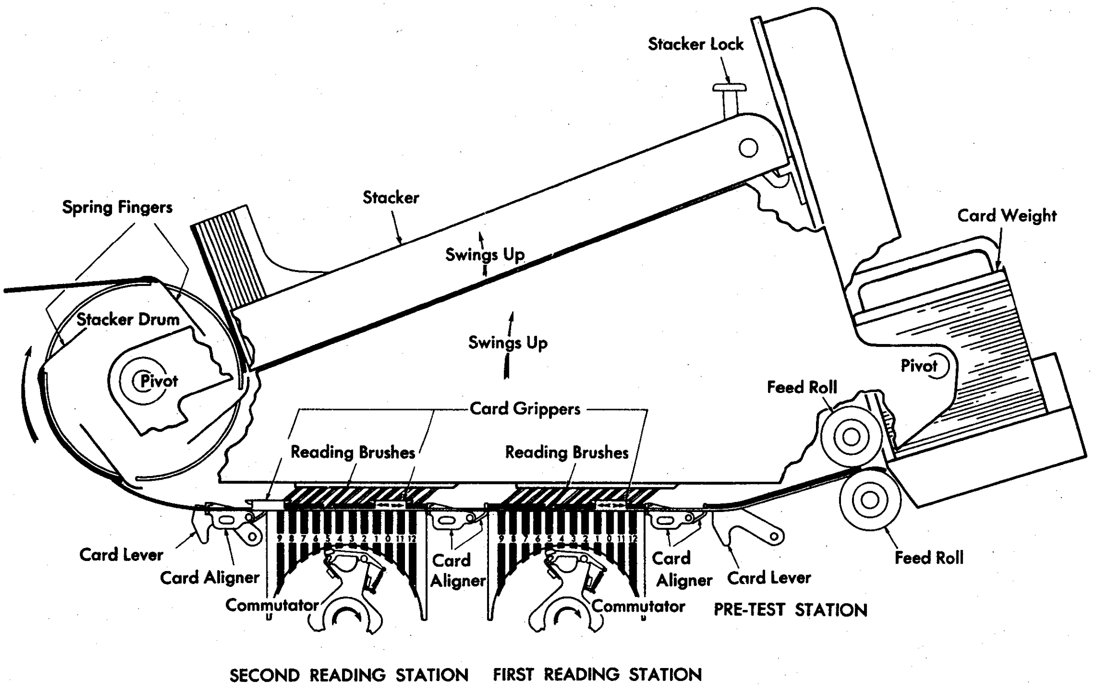


Figure 14.1 Feed unit, IBM 407 Accounting Machine.

ing station, one for every column of the card. The commutators rotate together, starting with the 9 position and advancing progressively to the 12 position. Impulses are transmitted to any brushes in contact with their corresponding segments through holes punched in a card. The brushes are connected internally to hubs on the control panel where punched-card data can be wired to control the various functions of the machine.

Because cards are read while stationary, any card can be read again and again. This feature permits more than one line to be printed from a single card. Also, amounts and quantities can be crossfooted from a single card into a counter for printing. Depending upon the model, the 407 speeds range from 150 lines per minute list and 150 lines tabulate to 50 lines list and 50 lines tabulate. Available counter capacity ranges from 168 positions down to 21 positions.

The 407 has 120 or 96 print wheels instead of typebars as shown in Figure 14.2. The wheels are arranged as though they rotated on a single shaft with their outside rims a fraction of an inch from the paper. The result is a bank of 120 possible printing positions within a width of 12 inches, or 12 characters to the inch. Each print wheel contains 47 separate character positions:

10 digits:	0 through 9
26 letters:	A through Z
11 special characters:	/ $ □ * % @ & – # . ,

As shown in Figure 14.2, the print wheel is divided into 12 equal parts corresponding to the arrangement of characters shown in Figure 14.3. Note that special characters are combinations of 8-3, 8-4 with the 0, 11, and 12 zone punches or no digits and single zone punches. In operation, the print wheels remain stationary until the numerical rows of the card are read. At this time, one of the corresponding 12 sections of the wheel is selected. A second selection of one of the four parts of a section is made when the zone rows are read. If no zone is read, the wheel is positioned to print the digit only.

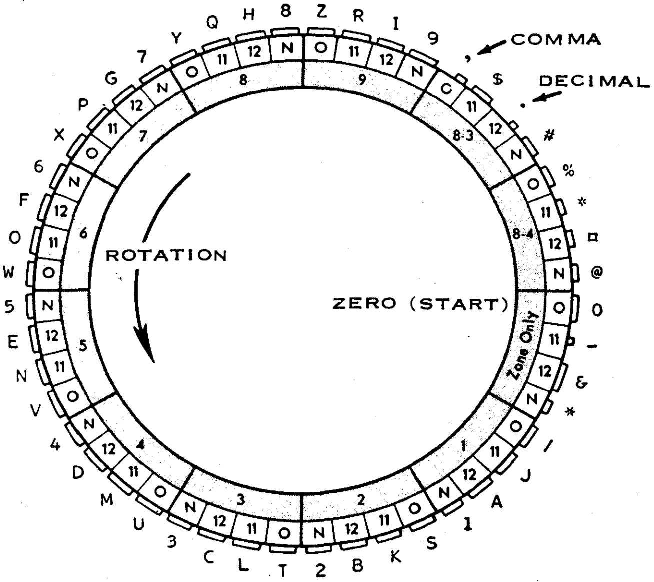

COMMA
DECIMAL
ROTATION
ZERO (START)
Zone Only
8
9
8-3
8-4
7
6
5
4
3
2
1
N
O
11
12
Q H 8 Z R I 9 , $. # % * □ @ 0 – & * / J A 1 S K B 2 T L C 3 U M D 4 V N E 5 W O F 6 X P G 7 Y

Figure 14.2 407 print wheel.

The print wheel rotates at a high rate of speed until printing time, when its speed is reduced to 25% of normal. At the actual time of printing, the wheel is moved against the platen in a straight line, producing maximum legibility. The rotary motion of the print wheel at print time is compensated by a special cam.

LOWER PUNCH	ZONE			
	12	11	O	N
1	A	J	/	1
2	B	K	S	2
3	C	L	T	3
4	D	M	U	4
5	E	N	V	5
6	F	O	W	6
7	G	P	X	7
8	H	Q	Y	8
9	I	R	Z	9
8-3	.	$	,	#
8-4	□	[1]*	%	@
	&	–	0	[2]*

1. Total Symbol 2. Check Protection

Figure 14.3 407 character arrangement.

Although one line is printed during one print cycle, the wheels print four different times within that cycle. All the wheels zoned for zero print first, followed in succession by those of 11, 12, and N.

Multiple Line Read Feature

The multiple line read (MLR) feature uses the capability of the 407 to read a card more than once. Each time a card is read, a line can be printed or a field can be entered into a counter. There is no limitation on the number of times one card can be read.

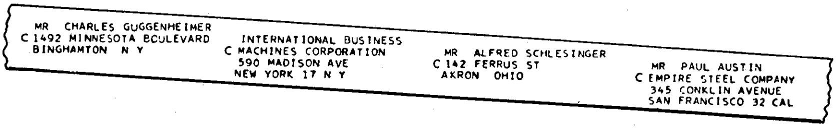
At least one column of the card must be set aside to identify the MLR card and then to determine the number of lines to be printed from that card. This is done by punching the letter A for one-line printing, the letter B for two-line printing, and C for three-line printing. In each case the 12 punch zone for the letters A, B, and C is used to start MLP operations. The 1-, 2-, and 3-digit punches of the letters are used to stop MLR operation after one, two, or three lines have been printed. More than three lines can be printed by making use of a *repeat* feature.

Normally, as many as 28 positions can be printed on a single line. If required, more than 28 can be printed by coupling coselectors.

The feature is particularly useful in applications of address printing. For example, address labels can be prepared on continuous forms. Interspersed three- and four-line addresses can be produced at rates that vary between 1,800 and 3,000 labels per hour. Specially designed forms, permitting multiple label printing side by side, increase production accordingly. After printing, the labels are separated and attached to the material to be mailed.

The 407 can be equipped with an address-writing feature to print three-line addresses at the rate of 9,000 per hour (or 150 per minute). Interspersed three- and four-line addresses can be produced at the rate of between 4,500 and 9,000 per hour.

In a high-speed application, addresses are printed on a seven-eighth inch tape from either 403 MLP cards or 407 MLR cards as shown in Figure 14.4. Over 7,000 carbon masters or 8,000 labels can be produced from one roll of tape. The carbon masters are used in commercial heat-transfer addressing machines and have a reverse carbon image printed on the back of the tape. Labels are processed by automatic mailing machines that cut and affix them to material for mailing.



MR CHARLES GUGGENHEIMER
C 1492 MINNESOTA BCULEVARD
BINGHAMTON N Y

INTERNATIONAL BUSINESS
C MACHINES CORPORATION
590 MADISON AVE
NEW YORK 17 N Y

MR ALFRED SCHLESINGER
C 142 FERRUS ST
AKRON OHIO

MR PAUL AUSTIN
C EMPIRE STEEL COMPANY
345 CONKLIN AVENUE
SAN FRANCISCO 32 CAL

Figure 14.4 Address printing on tape.

The address tape is fed by a special attachment that replaces the standard carriage of the machine. The tape travels across the platen roller at an angle. With this arrangement, four different lines of four separate addresses can be printed at the same time. Continuous form carbon paper passes face forward between the platen and the tape. Addresses are also printed on the face of the tape by normal print wheel and inked ribbon operation.

Printing is done from four fixed groups of print wheels. The right group prints the first line, the second group prints the second line, and the third group the third line. The fourth group at the extreme left prints only when a fourth line is required. Three-line addresses are punched in one card. Four-line addresses require two cards. A standard carriage tape is used to control the movement of the address tape across the platen.

Because the information must be printed from four different sources at the same time, the first line is printed from the first reading station. The second line is printed from the second reading station, and the third and fourth lines from counters and storage units.

A perforation is automatically cut in the address tape between each address as the tape passes from the platen to a rewind reel. These holes keep the tape in proper registration when it is processed by an automatic mailing machine.

Storage Units A 407 can be equipped with four 16-position storage units. The units store up to 64 positions of numerical information or 32 positions of alphabetic information from a card, from an emitter within the machine, or from a counter. Data can be read into or out of the storage units at will, under the control of a digit, an X punch, a card cycles impulse, a program exit, and certain types of impulses that are available from the operation of the automatic carriage. Each storage unit can store letters, digits, and three of the special characters: minus (—), ampersand (&), and diagonal (/). The remaining eight special characters include 8-3 and 8-4 numerical portions and therefore cannot be stored.

Storage units can be used as follows:

1. For storing information to be printed on sheet headings after a form overflow so that the name and address is duplicated on all subsequent forms of that group.

2. To increase the number of usable card columns for detail information by storing name, city, and state to be printed as required.

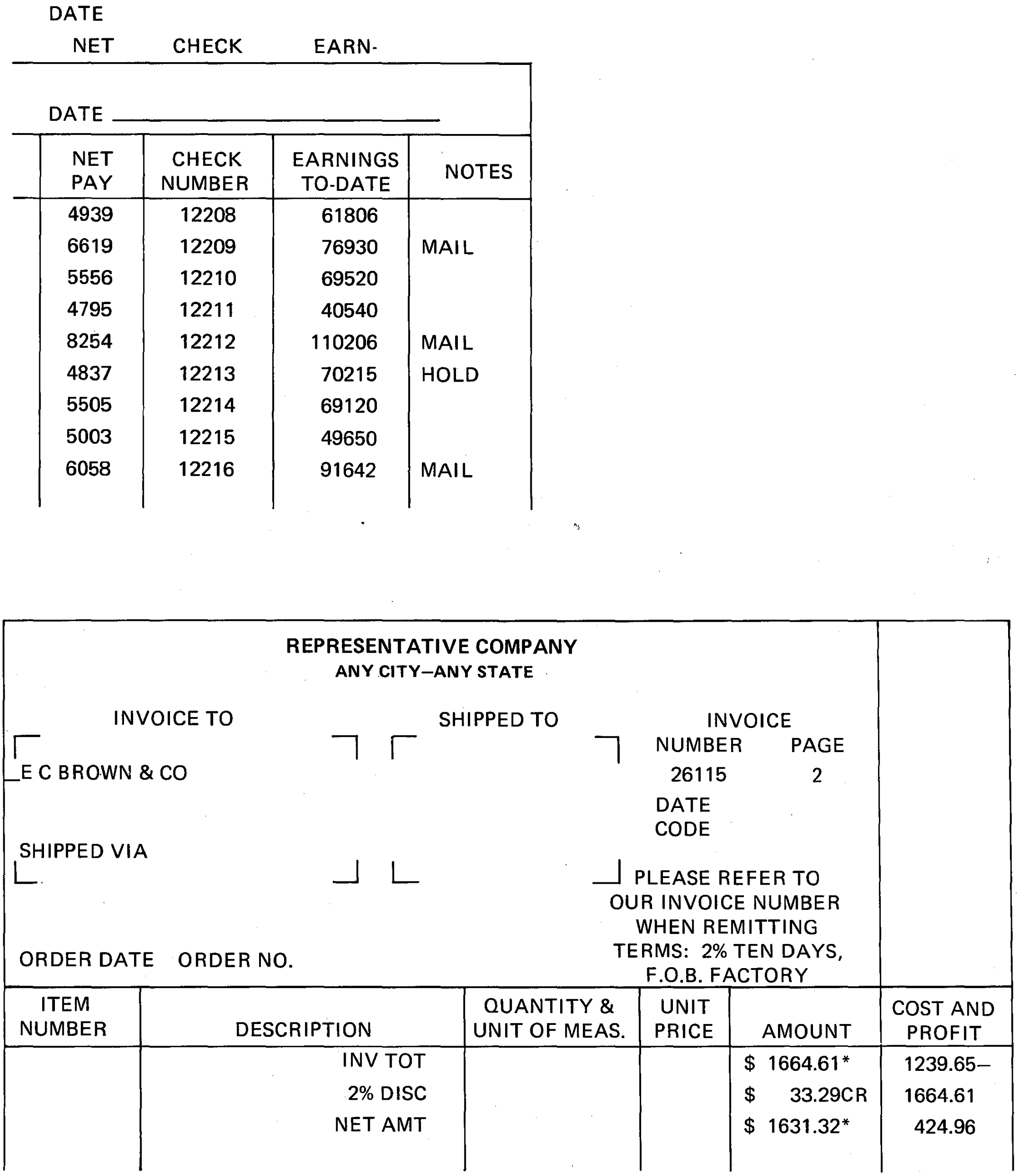
3. To store alphabetic or numerical information to be simultaneously read out and printed line by line with detail information from a card.

4. To store information to be summary punched.

5. To store information to be group indicated.

6. To store columnar heading information so blank unruled forms can be headed by reading out of storage for each new sheet.

Figure 14.5 indicates the use of information placed in storage for reporting printing.



DATE

NET CHECK EARN-

DATE ______________________

NET PAY	CHECK NUMBER	EARNINGS TO-DATE	NOTES
4939	12208	61806	
6619	12209	76930	MAIL
5556	12210	69520	
4795	12211	40540	
8254	12212	110206	MAIL
4837	12213	70215	HOLD
5505	12214	69120	
5003	12215	49650	
6058	12216	91642	MAIL

REPRESENTATIVE COMPANY
ANY CITY—ANY STATE

INVOICE TO
E C BROWN & CO

SHIPPED TO

INVOICE
NUMBER 26115
PAGE 2

DATE
CODE

SHIPPED VIA

PLEASE REFER TO
OUR INVOICE NUMBER
WHEN REMITTING
TERMS: 2% TEN DAYS,
F.O.B. FACTORY

ORDER DATE ORDER NO.

ITEM NUMBER	DESCRIPTION	QUANTITY & UNIT OF MEAS.	UNIT PRICE	AMOUNT	COST AND PROFIT
	INV TOT			$ 1664.61*	1239.65–
	2% DISC			$ 33.29CR	1664.61
	NET AMT			$ 1631.32*	424.96

Figure 14.5 Item and total labeling. Information in storage.

Tape-Controlled Carriage

The tape-controlled carriage feeds and spaces continuous paper forms at high speed for printing by the 407. The carriage is similar in operation to the carriages available with the 402 and 403 Accounting Machines previously described. The device can be programmed to control spacing, skipping, overflow, and form ejection by punching holes along a narrow tape that exactly corresponds in length to one or more forms. Once it is punched, the tape can be inserted in the carriage to control form operation and then removed to be used as often as required.

The carriage will accommodate forms up to a maximum width of 16 3/4 inches, including punched margins. Minimum width is 4 3/4 inches. The maximum length of the form depends upon the number of lines to the inch: 33 inches for a carriage that spaces four lines to the inch, 22 inches for six lines to the inch, and 16 1/2 inches for eight lines. While forms of any size within these limits can be handled, it is usually advisable to obtain forms of standard sizes from the manufacturers. Forms can also be designed to permit printing in practically any desired arrangement. Skipping can be controlled to 10 different sections of the form.

Single-, double-, or quadruple-spacing can vary between lines as controlled by wiring on the control panel. For example, the heading section of a form may be single-spaced and the body double-spaced. Any other spacing required must be controlled by the tape. Spaces up to two inches between lines can be skipped at the same rate as normal single-spacing. This skipping is a smooth, high-speed advance of the form, allowing successive lines to be printed up to two inches apart at the rate of 150 lines per minute.

The overflow punch in the tape can also be used to start other operations before ejecting a completely filled form. For example, a total may be printed at the bottom of each page before the form is advanced to the next sheet. Several lines of numerical or alphabetic identifying information can be printed on an overflow sheet. The information can be printed from storage. For example, invoice and page number of a multipage invoice can be printed at the top of each successive sheet.

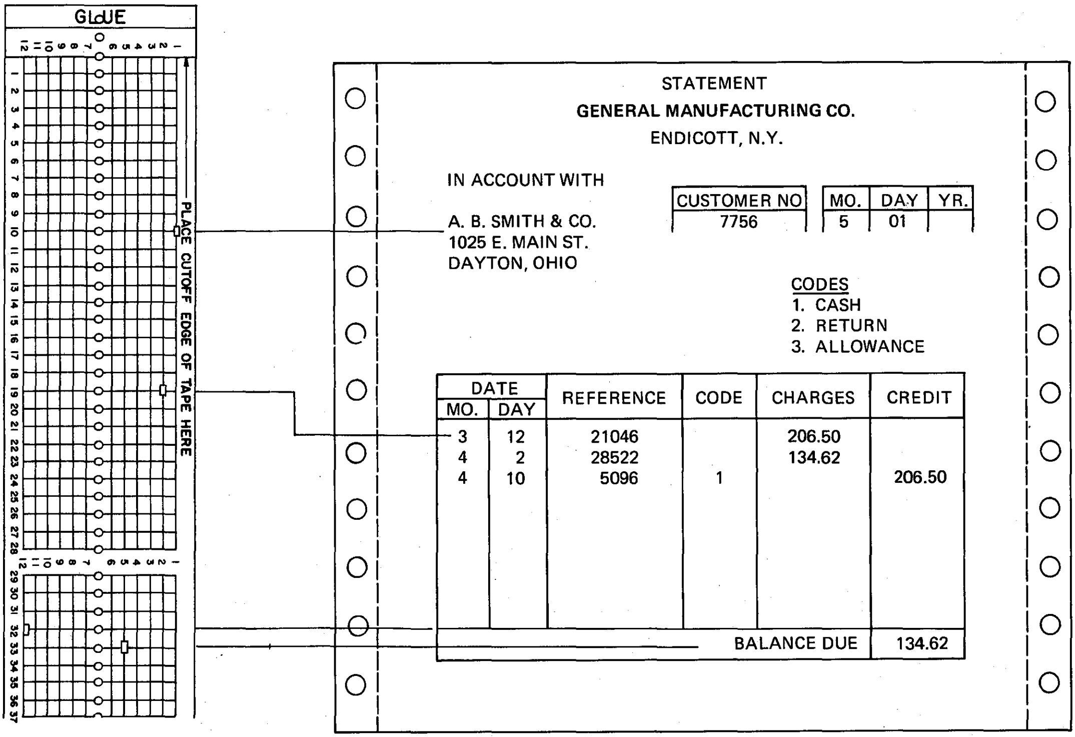
Totals can also be printed on a predetermined total line, whether the form is completely filled or not. For example, although only two or three items have been printed on a form, the total of these items may be printed on a previously designated line of the form, rather than directly beneath the last printed item.

When one sheet or page of a report is completely filled, it can be ejected and the next form can advance to either the first printing line or to the first body line. That is, spacing for the heading on the second sheet of an invoice differs from the spacing on the first sheet. Usually, printing on the second sheet starts directly on the miscellaneous data line with invoice and page number.

Overflow skipping is caused by sensing a punch in a specific position of the tape that starts advancing the paper to the required line of the next form. The carriage can be programmed so that if the last card of a group prints on the last available detail line, the accumulated total for that group prints before the form is skipped to the next sheet. Thus, the printing of a separate sheet for an unidentified total is prevented. Printing all totals on the last overflow form can be done without reducing the printing space on each of the preceding forms.

Control Tape

Figure 14.6 shows carriage control tape punching for predetermined printing locations. The tape is ruled for 12 columnar positions indicated by vertical lines. These positions are called *channels*. A maximum of 132 lines can be used for control of a



STATEMENT
GENERAL MANUFACTURING CO.
ENDICOTT, N.Y.

IN ACCOUNT WITH

A. B. SMITH & CO.
1025 E. MAIN ST.
DAYTON, OHIO

CUSTOMER NO	MO.	DAY	YR.
7756	5	01	

CODES
1. CASH
2. RETURN
3. ALLOWANCE

DATE		REFERENCE	CODE	CHARGES	CREDIT
MO.	DAY				
3	12	21046		206.50	
4	2	28522		134.62	
4	10	5096	1		206.50
				BALANCE DUE	134.62

Figure 14.6 Control punching for predetermined printing locations.

form, although for convenience the tapes are slightly longer. Horizontal lines are spaced six to the inch for the entire length of the tape. Round holes in the center are prepunched for a pinfeed drive in a tape sensing mechanism that controls the carriage. The tape advances through the mechanism in synchronism with the movement of the printed form through the carriage. The effect is exactly the same as if the control holes were punched along the edge of each form.

Twelve brushes, one for each channel, are positioned over the tape for sensing the punched holes. Brush 1 rests on channel 1, brush 2 on channel 2, and so on. A hole in the channel position allows the brush to make contact with a metal roller and set up the necessary circuits that can be used to control form operation. Carriage control hubs are available on the control panel. A special punch is available for punching the holes in the required positions in the channels. After the tape is punched it is cut and looped into a belt. The two ends are glued together. The completed tape is inserted in the carriage reading mechanism by the operator as part of the setup operation in preparation for a report run.

Form Design

One of the basic tools used in designing forms is the spacing chart, such as shown in Figure 14.7. The numbers across the top from 0 to 11 represent the tens position of the print wheel number. The numbers directly beneath represent the units position of the print wheel number. For example, print wheel 42 is located by referring first to the 4 column and then to the digit 2 within the 4 column. Print wheel 9 can be located by referring to the 0 column and then to the digit 9 within that column. The form alignment symbol (black triangle) locates print wheel 1, 60, or 120 and should be embodied in the form design as an aid to the operator in aligning the form when setting up the machine.

In this example of the spacing chart, a facsimile of the carriage control tape is shown at the left for marking the control punching for a specific form. Notations have been included concerning standard form widths and depths, lateral movement of the carriage, and instructions to forms manufacturers.

The following are some of the considerations that should be given to good form design on the 407:

1. Unlike the 402 and 403, every print wheel on the 407 contains all the characters that can be printed by the machine.

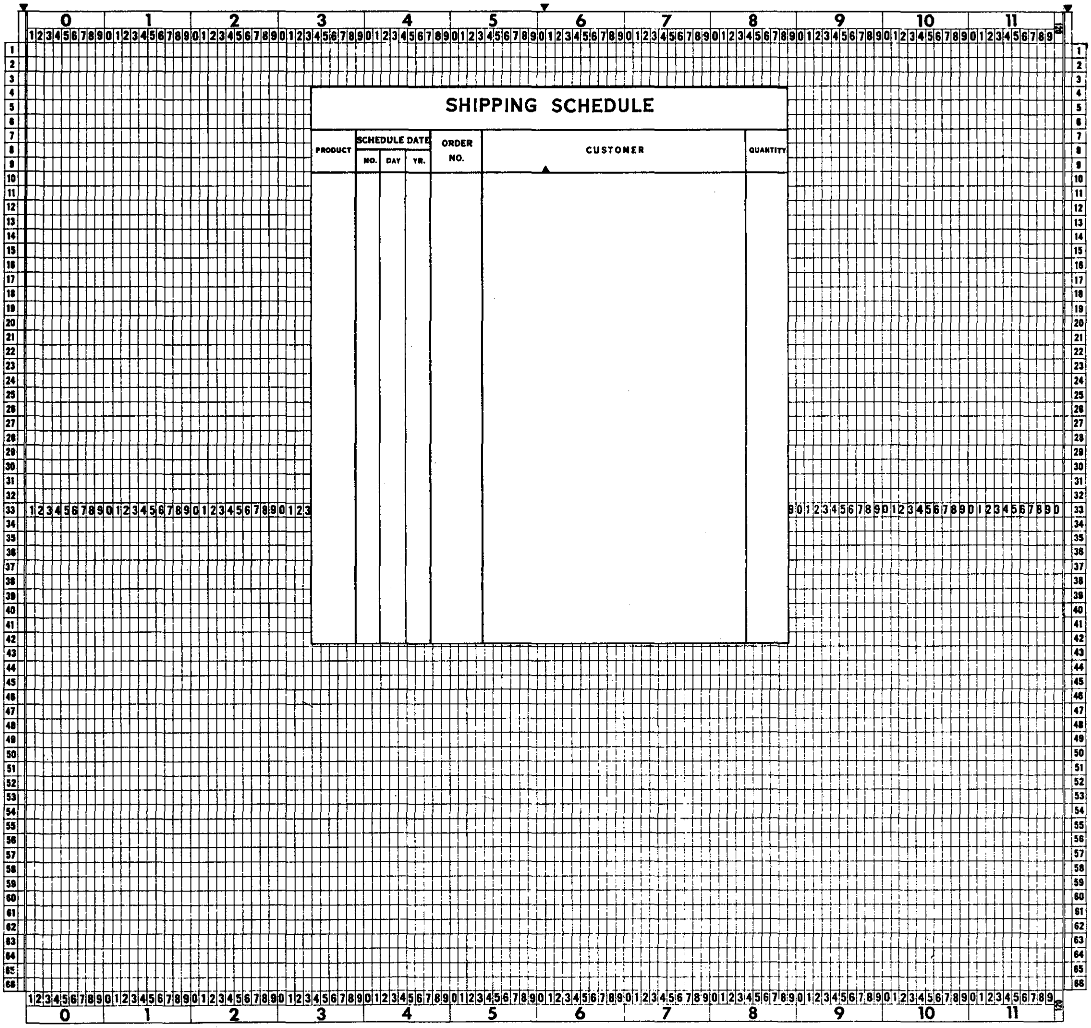


Figure 14.7 407 spacing chart.

Therefore, there is no restriction to position alphabetic information only to the left of the print area. Form space can be more efficiently utilized in some applications by taking advantage of this additional flexibility. For example, *sold to* and *ship to* names and addresses can be printed on the same line, one on the left side of the form and the other on the right. This is a particular advantage in the preparation of invoices, bills, orders, and the like. In this case, careful consideration must also be given to the header card design.

In some billing operations, the bill is made up in two parts: one an itemized statement and the other a remittance stub to be returned with payment. Customer name can be printed on both parts of the form by split wiring from the reading station to the print wheels.

2. The preprinting of vertical lines between two adjacent printing positions can be avoided because there is a maximum tolerance of only 0.010 inch between adjacent printed characters. The machine can be wired to print commas, decimals, oblique lines, dashes, etc., which can replace the vertical lines.

3. The number of legible multiple copies obtained depends upon the weight of the paper used for each form, the carbon coating, and the hardness of the platen. The striking force of the print wheels against the paper is not adjustable. Therefore, paper and carbon should be tested with a platen of the recommended hardness to be sure of best results.

4. On the 407, the credit symbol (CR) prints from two print wheels and the minus sign prints from one. For this reason, the minus sign is recommended as a credit symbol.

5. The dollar sign does not need to be preprinted on check forms. This symbol can be printed directly to the left of significant digits in the amount field. This *check protection* can be wired on the control panel to print amounts as $.50, $1.00, or $12.00, rather than $ .50, $ 1.00, or $ 12.00. The printing of the *floating dollar sign* by the machine prevents the printed amount from being fraudulently altered by filling in blank spaces in front of significant digits.

6. The number of alphabetic positions that can be assigned in a forms layout to print from storage is 32. More than 32 positions can be stored by combining the use of counters and co-selectors as required.

7. A *character emitter* can be wired through selectors or storage control to print report headings on unruled forms and to identify the first and last line of a report with such information as *brought forward* on the first line and *carried forward* on the last line.

8. A *variable length overflow* feature makes it unnecessary to reserve space for multiple total lines.

9. When spacing is six lines to the inch, skips of up to two inches are possible without interrupting the normal printing speed of the machine. Therefore, it is an important means of increasing machine production to hold forms to a size that does not require skipping in excess of these limits between print cycles.

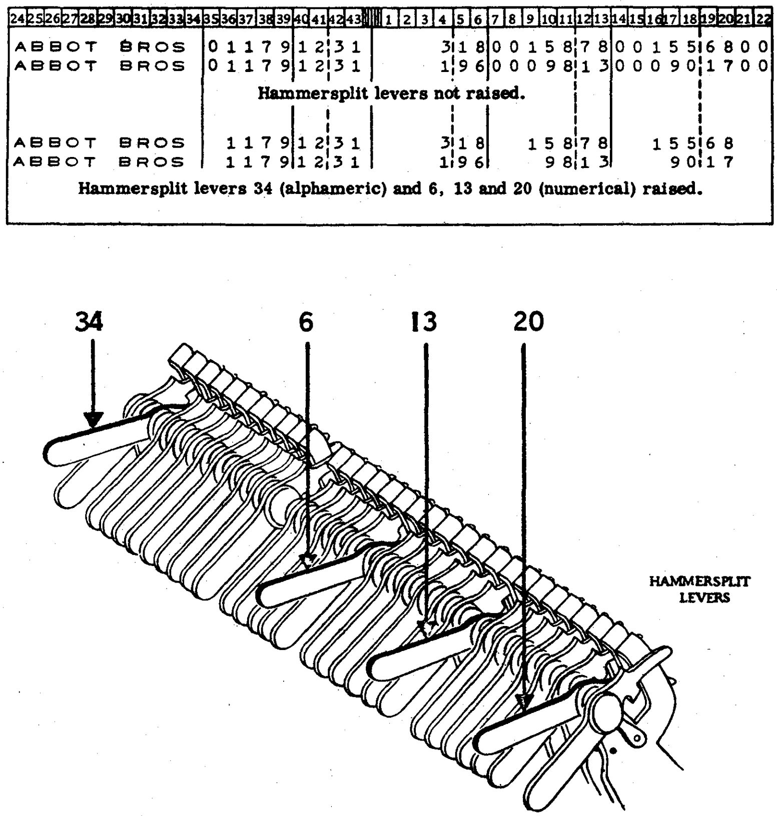
10. In the 402 and 403 print units, zero suppression to the right of a numerical field is mechanical. That is, unless certain levers are hand-set, zeros print automatically to the right of an amount field (Figure 14.8). Therefore, to suppress unwanted zeros, it is necessary to align all numerical fields to print in a columnar arrangement, including the total, thus:

```
Example:   (Customer number)    603918

                                   123
                                  1234
                                 12345
                                     1
                                    12
                           Total 13715*
```

Zero suppression on the 407 is electrical and completely flexible. It is not necessary to vertically align the right-hand digit of an amount field with the right-hand digit of a heading, for example, to print the customer number above an amount column. Also, total identification permits the flexible positioning of totals, thus:

```
Example:   (Customer number)    603918

                                  123
                                 1234
                                12345
                                    1
                                   12
                                          Total 13715*
```

Complete descriptions of all the operating features of tape-controlled carriages can be found in the manuals of operation supplied by the manufacturer. Complete forms specification for use with the carriage are also furnished by the forms manufacturers. These topics will not be covered in this text.



```
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 ||| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
ABBOT   BROS  0 1 1 7 9|1 2¦3 1          3¦1 8|0 0 1 5 8¦7 8|0 0 1 5 5¦6 8 0 0
ABBOT   BROS  0 1 1 7 9|1 2¦3 1          1¦9 6|0 0 0 9 8¦1 3|0 0 0 9 0¦1 7 0 0
              Hammersplit levers not raised.

ABBOT   BROS    1 1 7 9|1 2¦3 1          3¦1 8      1 5 8¦7 8      1 5 5¦6 8
ABBOT   BROS    1 1 7 9|1 2¦3 1          1¦9 6        9 8¦1 3        9 0¦1 7
   Hammersplit levers 34 (alphameric) and 6, 13 and 20 (numerical) raised.
```

Figure 14.8 402, 403 zero suppression.

Questions and Exercises

1. How does the 407 Accounting Machine differ from the 402 in card reading and feeding? Printing?
2. Describe the function of the 407 multiple line read feature.
3. What are the special design considerations for MLP cards?
4. What are storage units used for? How do they differ from counters?
5. What are the functions of the tape-controlled carriage?
6. Why is form design considerably more flexible on the 407 than on the 402?
7. What features of form design can be originated by the machine as reports are printed?
8. What is zero suppression?

9. What is a floating dollar sign?
10. Design the stationery expense reports from chapter 12 for printing on the 407.

15 INTRODUCTION TO COMPUTERS

Preceding chapters have described a number of data processing operations that can be effectively mechanized with specially designed machine equipment. For this type of automatic processing, the punched-card record has been presented as the standard medium of data handling and communication.

Operation	*Type of Equipment*
Transcribing original data into machine-readable records (punched cards)	Keypunch (80 column) Data recorder (96-column) Mark sensing punch (519)
Checking data transcription	Key verifier
Sorting and classification	Sorter
Reproducing and transferring data	Document-originating machine (519)
File handling	Collator
Calculating	Calculating punch
Report preparation	Accounting machine
Output card preparation	Summary punch

In all unit-record processing, the above operations are arranged in some predetermined sequence to form a planned procedure or *system*. Typically, machine records are punched, verified, sorted to sequence, associated with previous records, and then printed to report results. Each operation within the procedure is carried out on a separate machine unit specifically designed to perform that function. While this type of system is very effective when compared with many manual methods, some inherent disadvantages can quite readily be shown.

First: The sequence of operations is always under manual control. That is, to follow a given procedure, card records must be transported by hand from machine to machine. As a result, the system is entirely dependent upon the skill and ability of human operators to run the various machine units efficiently and to follow instructions correctly.

Second: Punched-card equipment is mainly electromechanical. Each operation requires physical movement of a paper record past reading or punching mechanisms. Therefore, processing has obvious limitations of speed. Production is still very fast by manual standards but nowhere near approaches the rates possible with electronic components.

To overcome these limitations, more advanced computing equipment has been designed and produced with the capability to perform all essential data processing operations in a single integrated unit. In such a computer system, electronic components are utilized in combination with electromechanical devices. Processing is almost entirely electronic while the operations of "reading and writing" records are performed with improved input and output devices interconnected through a *central processing unit.* The organization of a simple system is shown schematically in Figure 15.1.

As shown in the illustration, the input devices read directly from a record medium, in this case punched-card files. All information from each card is transmitted over a *channel* to a *storage unit* where the data is retained for processing. More than one input device can be provided, so that records from at least two different files can be read into storage at the same time; for example, a master record and associated details.

Before entering storage, input data is converted to a representation suitable for internal electronic manipulation. The entire system is controlled by a central processor made up of two sections: *control* and *arithmetic/logical*. The action of the processor is directed by a *stored program* also retained in the storage unit. Input data can be handled according to the procedure established by the program and then returned to storage where it is available for output. The completed output may take the form of either

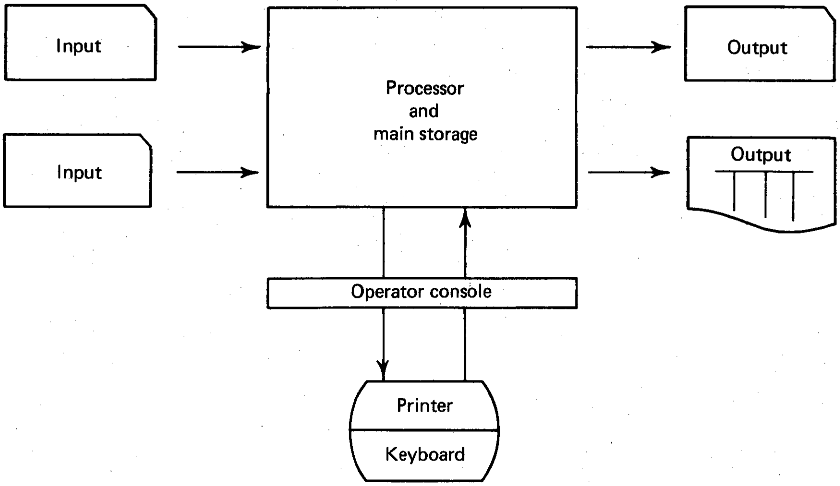


Figure 15.1 Schematic. Basic computer system.

punched cards, printed reports, or both. As data is "written" from storage, it is reconverted to punched-card code or to the graphic characters of printed information.

Limited input and output is also provided by a printer-keyboard. Operator control of the system is by means of a *console* where power is turned on or off, where the contents of storage can be displayed, and where system operation can be observed and monitored as required.

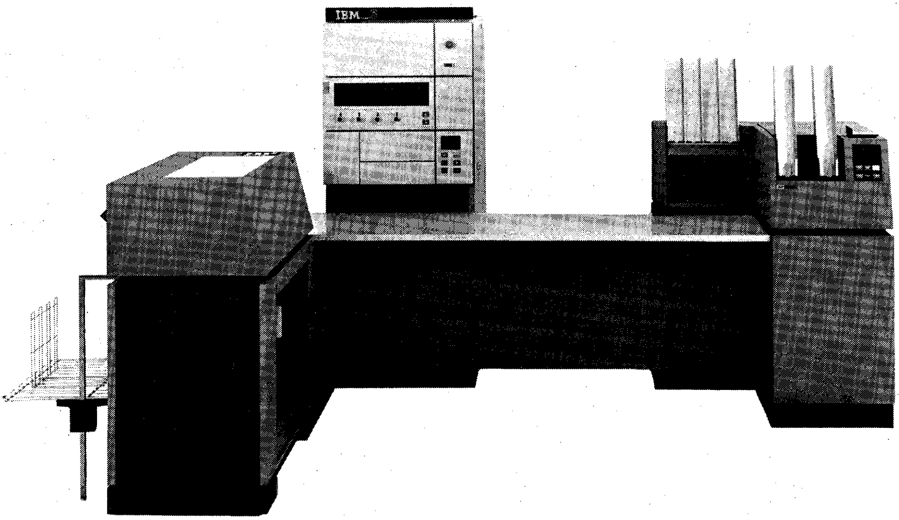


Figure 15.2 IBM System/3.

The following sections and Chapter 16 discuss each of these components in general terms as basic units common to all digital computers. Chapters 17 and 18 deal specifically with the IBM System/3, a small unit-record computer designed to operate with 96-column cards (Figure 15.2).

Computer Data Representation

It has been shown that data can be represented in many ways depending upon the medium of communication, recording, or storage, and the established conventions of coding for the medium. For example, in punched cards data is represented as patterns of holes. When transmitted to machines, the data is then represented by timed electrical pulses, the positions of counter wheels, and the relationships of various mechanisms within the device.

In a computer, data is represented by many electronic components: transistors, magnetic cores, electrical pulses, wires, and so on. The storage and flow of data through these devices is represented as electronic signals or indications. The presence or absence of these signals in specific circuitry is the method of representing data, much as the presence or absence of holes in a card represents data.

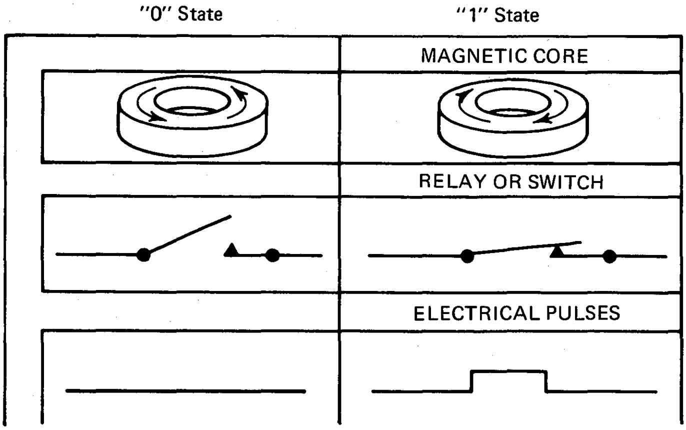


Figure 15.3 Binary state of computer components.

Computers function in binary states; that is, computer components can indicate only two possible states or conditions. For example, the ordinary light bulb operates in a binary mode – it is either on or off. Likewise within the computer – transistors are maintained either conducting or nonconducting, magnetic materials are magnetized in one direction or in the opposite direction, and specific voltage potentials are present or absent. These conditions are illustrated in Figure 15.3. The binary state of the various components is a signal to the computer, just as the presence or absence of light from an electric bulb can be a signal to a person.

For example, a device to represent decimal values can be designed with four electric bulbs and with switches to turn each bulb on or off as shown in Figure 15.4. The light bulbs are given decimal values of 8, 4, 2, and 1. When a bulb is lighted, it represents the decimal value associated with it. When a bulb is not lighted, no value is represented. With such an arrangement, the total decimal value will be the numeric sum indicated by the lighted bulbs.

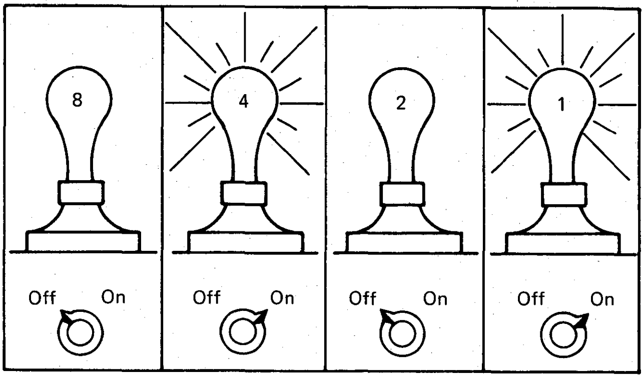


Figure 15.4 Binary coded decimal representation.

All decimal values 0 through 15 can be represented. The value 0 is represented by all lights off; the value 15 by all lights on. The value 9 is represented by having the 8 and 1 lights on and the 4 and 2 lights off. The value 5 is represented by having the 4 and 1 lights on and the 8 and 2 lights off, and so on.

Note that the bulbs may be given almost any significance, depending upon what mode of operation is to be established. For example, the 16 possible on/off combinations can represent letters of the alphabet. When the right-hand bulb is on, assume that it represents the letter A. When the second bulb from the right is on, assume it represents the letter B. When both bulbs are on, the letter C is represented, and so on. Because there are 26 letters in the alphabet, one more bulb must be added to the device to represent all letters.

Data is represented within a computer by assigning a specific value or significance to a single binary indication or group of binary indications. The significance assigned to the specific indications becomes the code or *language* for representing data. Calculations can also be performed using the binary number system and binary arithmetic.

Many data processing operations are logical in nature. Since logic is fundamentally a yes/no system, these operations can be performed very efficiently in binary. For example, when one value is compared against another, the result of the comparison can be indicated as high or low, equal or not equal. When a calculation is performed, the result is nearly always indicated as plus or minus, positive or negative, debit or credit.

Many control operations within the machine system can also be expressed in the binary state. Is the card reader ready to feed the next card? Yes or no? Is an error indicated in the transmission of input data to storage? Yes or no? Does the code structure of the data received conform to the accepted convention of the system? Yes or no?

In computer parlance, the binary state is often referred to as the presence or absence of a "bit." Probably the term originated to describe the recording on magnetic tape where combinations of magnetized "bits" and "no-bits" in the surface of the recording material are used to represent data. As will be shown in the next section, these terms have carried over into references commonly made to the ones and zeros of binary notation and to computer codes.

Binary Number Systems

The design and programming of an application for computers does not normally involve any direct use of an internal machine language. It will be shown later that *programming systems* have been developed which make it possible to write instructions for a computer in a language much more convenient to use than direct binary representation. The computer itself can convert from this programming language to its own internal system of coding.

However, to understand the actual capability of the machine, including storage capacity, operating characteristics, and so on, it is helpful to have some general knowledge of the various machine number systems.

The familiar decimal (base 10) number system uses 10 symbols or digits to represent numeric values. The symbols are 0 through 9. The place value of the digits also signifies units, tens, hundreds, thousands, etc.

The binary (base 2) number system uses only two symbols or digits: 0 and 1. The place value of the symbols is based on the progression of the powers of 2. That is, the units position of a binary number has a value of 1; the next position, a value of 2; the next, 4; the next, 8; the next, 16; and so on. These and additional place values are shown in Figure 15.5.

8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

Figure 15.5 Place values of binary numbers.

In pure binary notation, the digits indicate whether the corresponding power of 2 is present or absent in each position of the number. A 1 represents the presence of the number; the 0 represents the absence of the value. The place value does not signify units, tens, hundreds, or thousands as in the decimal system. Instead, the place value signifies units, twos, fours, eights, sixteens, and so forth. In this system, the quantity 12 is expressed with the symbols 1100, meaning $(1 \times 2^2) + (1 \times 2^2) + (0 \times 2^1) + (0 \times 2^0)$ or $(\mathit{1} \times \mathit{8}) + (\mathit{1} \times \mathit{4}) + (\mathit{0} \times \mathit{2}) + (\mathit{0} \times \mathit{1})$. Figure 15.6 shows the binary representations of the decimal values 0 through 16.

Computers using the binary system of data representation are typified by the IBM System/3, System/360, and System/370. For all of these systems, the basic unit of information is a *byte*, made up of 8 binary positions. Four bytes make a computer *word* with 32 consecutive bit positions of information which are interpreted as a unit, much as a character of digit in other systems.

The bit sections within a word have a place significance related to the binary number system. That is, the place position of a bit in the word determines the value of the bit. In the binary number system, the decimal values of the places (from right to left) are 1, 2, 4, 8, 16, 32, 64, and so on.

Although the place value of the bits of a word are always those of the binary number system, they can be interpreted or processed in such a way as to represent a significance other than quantity. For example, a 32-bit word can be interpreted as a 32-place binary number, as an 8-digit *hexadecimal* number, as four alphabetic or numeric characters, or as any predetermined representation.

Decimal Value	Place Value				
	16	8	4	2	1
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	0	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	0	1	0	0	1
10	0	1	0	1	0
11	0	1	0	1	1
12	0	1	1	0	0
13	0	1	1	0	1
14	0	1	1	1	0
15	0	1	1	1	1
16	1	0	0	0	0

Figure 15.6 Binary equivalent of decimal values for 1 through 16.

Octal Number System

It is apparent that binary numbers require several times as many positions as decimal numbers to display the equivalent number. In talking and writing, these binary numbers are awkward. A long string of ones and zeros cannot be effectively transmitted from one individual to another. The octal and hexadecimal number systems fill this need. Because of their simple relation to the binary number system, numbers can be converted from one system to another by inspection.

The base or *radix* of the octal system is 8. This means that there are eight symbols to represent all values. The symbols are 0 through 7; there are no symbols 8 and 9. The important relationship is that three binary positions are equivalent to one octal position. The sample chart in Figure 15.7 is used to convert between binary, octal, and decimal systems.

Decimal	*Binary*	*Octal*
0	000 000	0
1	000 001	1
2	000 010	2
3	000 011	3
4	000 100	4
5	000 101	5
6	000 110	6
7	000 111	7
8	001 000	10
9	001 001	11
10	001 010	12
11	001 011	13
12	001 100	14
13	001 101	15
14	001 110	16
15	001 111	17
16	010 000	20

Figure 15.7 Binary and octal equivalent of decimal values for 1 through 16.

The octal system is used mainly to provide a shorthand. The digits of the number represent the coefficients of the ascending powers of 8. Consider the octal number:

$$\begin{aligned} 173 &= (1 \times 8^2) + (7 \times 8^1) + (3 \times 8^0) \\ &= \quad 64 \quad + \quad 56 \quad + \quad 3 \\ &= 123 \text{ (decimal)} \end{aligned}$$

Integer Conversion

By remembering what a number represents in the binary or octal system, the number can be converted to its decimal equivalent by the method shown above. However, with larger numbers, this method becomes more difficult to use. More convenient methods are described in the following sections.

Decimal to Octal

Rule: Continuously divide the decimal number by 8 and develop the octal number from the remainders of each step in the division.

Example: Convert the decimal number 149 to its octal equivalent.

```
8 |149  remainder 5
8   |18  remainder 2
8     |2  remainder 2
       0
```

The octal equivalent of decimal 149 is 225.

The original number to be converted is first divided by 8 as shown above. The remainder from this division becomes the low-order digit of the octal number, or 5. The quotient received from the first division is then divided by 8. The remainder becomes the next higher-order position of the octal number, or 2. The steps are repeated until the quotient is smaller than the divisor. After this step, the final quotient is considered to be the high-order digit of the conversion; in this example, 2.

Octal to Decimal

Rule: Continuously multiply by 8 and then add the next octal digit.

Example: Convert the octal number 225 to its decimal equivalent.

```
   2       2       5
  X8       |       |
 ----      |       |
  16       |       |
  +2 <-----        |
 ----              |
  18               |
  X8               |
 ----              |
 144               |
  +5 <-------------
 ----
 149
```

The high-order digit of the octal number is multiplied by 8 and the next lower-order digit is added to the product. The result is then multiplied by 8, and the next lower-order octal digit is added to that amount. When the lowest-order octal digit has been added, the process ends. The decimal equivalent of octal 225 is shown to be 149.

Octal to Binary and Binary to Octal

This can be done visually with no calculation required.

Rule: Express the number in binary groups of three.

Examples: Octal to Binary

$$\underbrace{2}_{010}\quad\underbrace{2}_{010}\quad\underbrace{5}_{101}$$

Binary to Octal

$$\underbrace{010}_{2}\quad\underbrace{010}_{2}\quad\underbrace{101}_{5}$$

Decimal to Binary

Rule: Continuously divide the decimal number by 2 and develop the binary number from the remainders.

Example: Convert decimal 149 to its binary equivalent.

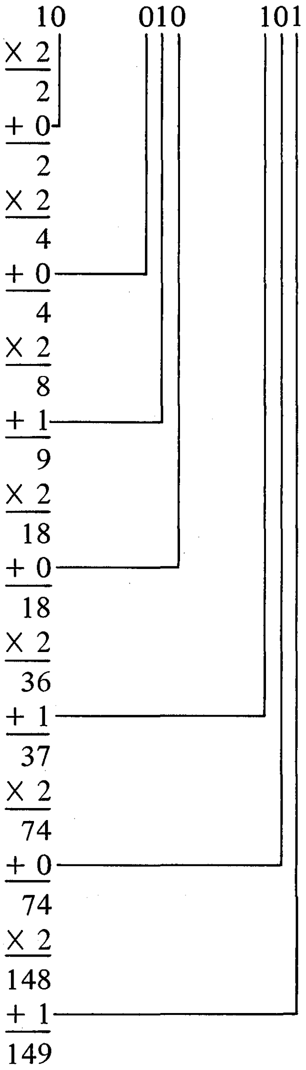
$2\,\underline{|149}$ remainder 1
$2\,\underline{|74}$ remainder 0
$2\,\underline{|37}$ remainder 1
$2\,\underline{|18}$ remainder 0
$2\,\underline{|9}$ remainder 1
$2\,\underline{|4}$ remainder 0
$2\,\underline{|2}$ remainder 0
$2\,\underline{|1}$ remainder 1
0 remainder 0

Arrange the binary number as 010 010 101, reading from the bottom of the column to the top.

Binary to Decimal

Rule: Continuously multiply the binary digits by 2 and add the next binary digit.

Example: Convert 010 010 101 to its decimal equivalent.



The decimal equivalent of 010 010 101 is shown to be 149.

Fraction Conversion

Fractions can also be converted from one number system to another as shown in the following examples.

Decimal to Octal

Rule: Continuously multiply by 8 and develop the octal number from the carry.

Example:

$$\begin{array}{r} .149 \\ \times\quad 8 \\ 1.192 \\ \times\quad 8 \\ 1.536 \\ \times\quad 8 \\ 4.288 \\ \times\quad 8 \\ 2.304 \end{array}$$

The octal equivalent of the decimal fraction .149 is shown to be 0.1142+.

Octal to Decimal

Rule: Express as powers of 8, and add.

Example:

$$\begin{aligned} 0.1142 &= 1(8^{-1}) + 1(8^{-2}) + 4(8^{-3}) + 2(8^{-4}) \\ &= \frac{1}{8} + \frac{1}{64} + \frac{4}{512} + \frac{2}{4096} \\ &= 0.1489 \text{ or } 0.149 \end{aligned}$$

The decimal equivalent of the octal fraction 0.1142 is shown to be 0.149.

Octal to Binary and Binary to Octal

Rule: The same rule applies for fractions as for whole numbers.

Example:

$$\begin{array}{llll} .1 & 1 & 4 & 2 \\ .001 & 001 & 100 & 010 \end{array}$$

$$\begin{array}{llll} .001 & 001 & 100 & 010 \\ .1 & 1 & 4 & 2 \end{array}$$

Binary to Decimal

Rule: The same rule applies as for whole numbers.

Example: .001 001 100 010

$$= 1(2^{-3}) + 1(2^{-6}) + 1(2^{-7}) + 1(^{-11})$$

$$= \frac{1}{8} + \frac{1}{64} + \frac{1}{128} + \frac{1}{2048}$$

$$= 305/2048$$

$$= .1489 \text{ or } .149$$

Hexadecimal Number System

While many computers use the binary number system to indicate the power of 2 up to the size of a full word or designated field, any direct man-machine communication is in the hexadecimal numbering system. Large binary numbers are made up of long strings of ones and zeros, frequently awkward to interpret and handle. The hexadecimal (base 16) numbering system is used as a convenient way to represent such large binary numbers. Each hexadecimal digit stands for four binary digits.

Hexadecimal notation requires 16 symbols to represent 16 number values. Since the decimal system provides only 10 number symbols (0 through 9), six additional symbols are needed to represent the remaining values. The letters A, B, C, D, E, and F have been adopted for this purpose, though any other six marks could have served equally well. Thus, the entire list of hexadecimal symbols consists of 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F, in ascending sequence of value. The following table shows equivalent decimal, binary, and hexadecimal numbers.

Decimal	*Hexadecimal*	*Binary*
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001

Decimal	*Hexadecimal*	*Binary*
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111
16	10	10000
17	11	10001
18	12	10010
19	13	10011
20	14	10100
21	15	10101
22	16	10110
23	17	10111
24	18	11000
25	19	11001
26	1A	11010
27	1B	11011
28	1C	11100
29	1D	11101
30	1E	11110
31	1F	11111

Note that upon reaching decimal 16, the hexadecimal symbols are exhausted, and a "carry" is placed in front of each hexadecimal symbol. This carry symbol (1) has a place value of decimal 16.

Binary to Hexadecimal

Rule: Divide the binary number into groups of four binary digits, starting from the right, and replace each group with the corresponding hexadecimal symbol. If the left-hand group is incomplete, fill in zeros as required.

Example:

```
111110011011010011 = 0011  1110  0110  1101  0011
                   =  3     E     6     D     3
                   or 3E6D3 (hexadecimal)
```

Hexadecimal to Binary

Rule: Substitute the corresponding group of four binary digits for each hexadecimal symbol and drop off any unnecessary zeros.

Example: 6C4F2E (hexadecimal)

= 6 C 4 F 2 E

= 0110 1100 0100 1111 0010 1110

= 011011000100111100101110 (binary)

Integer Conversion, Hexadecimal to Decimal

Converting large hexadecimal numbers by either direct expansion or a multiplication-addition method becomes quite tedious and difficult. The use of conversion tables makes possible rapid conversion of hexadecimal integers into equivalent decimals, so that necessary arithmetic can be performed in the decimal system.

	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	268,435,456	1	16,777,216	1	1,048,576	1	65,536	1	4,096	1	256	1	16	1	1
	2	536,870,912	2	33,554,432	2	2,097,152	2	131,072	2	8,192	2	512	2	32	2	2
	3	805,306,368	3	50,331,648	3	3,145,728	3	196,608	3	12,288	3	768	3	48	3	3
	4	1,073,741,824	4	67,108,864	4	4,194,304	4	262,144	4	16,384	4	1,024	4	64	4	4
	5	1,342,177,280	5	83,886,080	5	5,242,880	5	327,680	5	20,480	5	1,280	5	80	5	5
	6	1,610,612,736	6	100,663,296	6	6,291,456	6	393,216	6	24,576	6	1,536	6	96	6	6
	7	1,879,048,192	7	117,440,512	7	7,340,032	7	458,752	7	28,672	7	1,792	7	112	7	7
	8	2,147,483,648	8	134,217,728	8	8,388,608	8	524,288	8	32,768	8	2,048	8	128	8	8
	9	2,415,919,104	9	150,994,944	9	9,437,184	9	589,824	9	36,864	9	2,304	9	144	9	9
	A	2,684,354,560	A	167,772,160	A	10,485,760	A	655,360	A	40,960	A	2,560	A	160	A	10
	B	2,952,790,016	B	184,549,376	B	11,534,336	B	720,896	B	45,056	B	2,816	B	176	B	11
	C	3,221,225,472	C	201,326,592	C	12,582,912	C	786,432	C	49,152	C	3,072	C	192	C	12
	D	3,489,660,928	D	218,103,808	D	13,631,488	D	851,968	D	53,248	D	3,328	D	208	D	13
	E	3,758,096,384	E	234,881,024	E	14,680,064	E	917,504	E	57,344	E	3,584	E	224	E	14
	F	4,026,531,840	F	251,658,240	F	15,728,640	F	983,040	F	61,440	F	3,840	F	240	F	15
Hexadecimal Positions	8		7		6		5		4		3		2		1	

Figure 15.8 Hexadecimal–decimal integer conversion.

Figure 15.8 is a hexadecimal/decimal integer conversion table. Each column of the table is labeled *at the bottom* with the number of the hexadecimal place position from *right* to *left*. Each column also has all the hexadecimal symbols arranged in ascending order of values from top to bottom with the corresponding decimal equivalent. Thus, the hexadecimal symbol D in column 1 has an equivalent decimal value of 13; in column 2 the equivalent value is 208; in column 4 the equivalent is 53,248; and so on.

Rule: Add the decimal equivalents of the hexadecimal digits, starting with the units position of the hexadecimal number. Work from right to left, using the table for the decimal equivalents of the higher-order positions.

Example: Convert hexadecimal A4B5 to decimal.

5 in hex position 1 =	5
B in hex position 2 =	176
4 in hex position 3 =	1,024
A in hex position 4 =	40,960
The sum = decimal value	42,165

Integer Conversion, Decimal to Hexadecimal

While not quite so convenient, it is possible to convert from decimal to hexadecimal using the conversion table as Figure 15.7.

Rule: Look up in the table the next smaller number than the decimal number to be converted. Note the hex equivalent and position number. Subtract the decimal value of that hex number from the decimal number and then look up the remainder in the table. Repeat the process.

Example: Convert the decimal number 42,165 to hexadecimal.

Find the hex equivalent of decimal	= 42,165
A in position 4	= 40,960
Remainder	= 1,205
4 in position 3	= 1,024
Remainder	= 181
B in position 2	= 176
Remainder	= 5
5 in position 1	= 5

The hexadecimal equivalent of decimal 42,165 is shown to be A4B5.

Questions and Exercises

1. List the normal sequence of steps in a punched-card data processing procedure.
2. What are the essential elements of a computer system?
3. How can values, characters, or other data be represented by binary devices?
4. What other types of logical operations can be conveniently represented by binary devices?
5. What is a byte? What can it represent?
6. Write the binary and octal equivalents of the following decimal numbers: 3, 5, 7, 9, 12, 15, 21.

7. Convert the following decimal numbers to octal: 125, 19, 288, 966, 1346.
8. Convert the following octal numbers to decimal and binary: 437, 16, 114, 525, 50.
9. Convert from binary to decimal: 101111101, 111101010.
10. Using the table in Figure 15.8, convert the following hexadecimal numbers to decimal: 4A5B, 1CD, 7EF, E6B.
11. Using the table in Figure 15.8, convert the following decimal numbers to hexadecimal: 24681, 4832, 1234.

16 COMPUTER CODES AND MAIN COMPONENTS

Data can be symbolized visually in many ways, including the most common method of using printed characters and digits to record information. In the computer, as in punched-card machines, data cannot be seen as printed information but must be represented or coded with a fixed number of binary indications. For example, a code can be constructed to represent all numeric and alphabetic characters with eight positions of binary indications made up of yes/no, on/off, or 0 and 1 bit combinations in some electronic form.

Most computer codes are self-checking; that is, they are constructed with some method of checking the validity of the coded representation. The code checking is designed to occur automatically within the machine as normal data processing operations are carried out. The method of validity checking is a part of the code design.

In some codes, each unit or character of data is represented by a specific number of bit positions that must always contain an even number of 1 bits. Different characters are made up of different combinations of 0 and 1 bits, but the number of 1 bits in any valid character is always even. With this code system, a character with an odd number of 1 bits is detected as an error. Likewise, a code may be used in which all characters must have an odd number of 1 bits. In this code, an error is indicated if a character is detected with an even number of 1 bits.

This type of checking is known as a *parity* check. Codes that require an even number of 1 bits are said to have *even* parity. Codes that require an odd number of 1 bits are said to have *odd* parity.

In other codes, the number of 1 bits present in each unit of data is fixed. For example, a code may use eight bit positions to code all characters, but exactly four 1 bits will be present in each character. Characters having more or fewer than four 1 bits cause an error indication. This system of checking is known as a *fixed-count* check and is often used for data transmission in teleprocessing networks where computers may receive and send data over communication lines.

Binary Coded Decimal (BCD)

In this code, all characters, including numeric, alphabetic, and special, are represented with six positions of binary notation plus a parity bit position. These positions are divided into three groups: one check position, two zone positions, and four numeric positions. The code structure is shown in Figure 16.1.

		0	1	2	3	4	5	6	7	8	9
Check	C	0	1	1	0	1	0	0	1	1	0
Zone	B	0	0	0	0	0	0	0	0	0	0
	A	0	0	0	0	0	0	0	0	0	0
Numerical	8	1	0	0	0	0	0	0	0	1	1
	4	0	0	0	0	1	1	1	1	0	0
	2	1	0	1	1	0	0	1	1	0	0
	1	0	1	0	1	0	1	0	1	0	1

		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Check	C	1	1	0	1	0	0	1	1	0	0	0	1	0	1	1	0	0	1	0	1	0	1	1	0	0	1
Zone	B	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
	A	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Numerical	8	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1
	4	0	0	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
	2	0	1	1	0	0	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
	1	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

		&	.	□	−	$	*	/	,	%	#	@
Check	C	0	1	0	1	0	1	0	0	1	1	0
Zone	B	1	1	1	1	1	1	0	0	0	0	0
	A	1	1	1	0	0	0	1	1	1	0	0
Numerical	8	0	1	1	0	1	1	0	1	1	1	1
	4	0	0	1	0	0	1	0	0	1	0	1
	2	0	1	0	0	1	0	0	1	0	1	0
	1	0	1	0	0	1	0	1	1	0	1	0

Figure 16.1 Binary coded decimal.

The four numeric positions are assigned the decimal values 8, 4, 2, and 1, and can represent the numeric digits 0 through 9. Note that the digit 0 is arbitrarily represented as 1010, actually the binary number for 10. The B and A zone bits are 0 when only the numeric digits are represented.

Combinations of zone and numeric bits represent alphabetic and special characters. The B and A bits provide for four possible bit combinations: 01, 10, 11, and 00, or the binary notation for decimal 1, 2, 3, and 0. The code structure follows that of 80-column cards, that is, three zones plus numeric values represent letters. No zones with numeric values represent digits.

The C position is the check bit used for code checking only. Because the six-bit alphameric code is an even parity code, the total number of bits used to represent a character must include

	CHARACTER		CARD CODE	BCD CODE (Core Storage)						
	Report	Program								
Low →	b		No Punches	C						
	•		12-3-8		B	A	8		2	1
	□	)	12-4-8	C	B	A	8	4		
	[		12-5-8		B	A	8	4		1
	<		12-6-8		B	A	8	4	2	
	⧧		12-7-8	C	B	A	8	4	2	1
	&	+	12	C	B	A				
	$		11-3-8	C	B		8		2	1
	*		11-4-8		B		8	4		
	]		11-5-8	C	B		8	4		1
	;		11-6-8	C	B		8	4	2	
	Δ		11-7-8		B		8	4	2	1
	–		11		B					
	/		0-1	C		A				1
	,		0-3-8	C		A	8		2	1
	%	(	0-4-8			A	8	4		
	ᵔ		0-5-8	C		A	8	4		1
	\		0-6-8	C		A	8	4	2	
	⧻		0-7-8			A	8	4	2	1
	ƀ		2-8			A				
	#	=	3-8				8		2	1
	@	'	4-8	C			8	4		
	:		5-8				8	4		1
	>		6-8				8	4	2	
	√		7-8	C			8	4	2	1
	?		12-0	C	B	A	8		2	
	A		12-1		B	A				1
COLLATING SEQUENCE	B		12-2		B	A			2	
	C		12-3	C	B	A			2	1
	D		12-4		B	A		4		
	E		12-5	C	B	A		4		1
	F		12-6	C	B	A		4	2	
	G		12-7		B	A		4	2	1
	H		12-8		B	A	8			
	I		12-9	C	B	A	8			1
	!		11-0		B		8		2	
	J		11-1	C	B					1
	K		11-2	C	B				2	
	L		11-3		B				2	1
	M		11-4	C	B			4		
	N		11-5		B			4		1
	O		11-6		B			4	2	
	P		11-7	C	B			4	2	1
	Q		11-8	C	B		8			
	R		11-9		B		8			1
	‡		0-2-8			A	8		2	
	S		0-2	C		A			2	
	T		0-3			A			2	1
	U		0-4	C		A		4		
	V		0-5			A		4		1
	W		0-6			A		4	2	
	X		0-7	C		A		4	2	1
	Y		0-8	C		A	8			
	Z		0-9			A	8			1
	Ø		0	C			8		2	
	1		1							1
	2		2						2	
	3		3	C					2	1
	4		4					4		
	5		5	C				4		1
	6		6	C				4	2	
	7		7					4	2	1
	8		8				8			
High →	9		9	C			8			1

NOTE: Tape may use even parity.

Figure 16.2 BCD interchange code.

an even number of 1 bits or the character is considered invalid. A check bit 1 is added whenever the sum of the numeric and zone bits is odd. If the number of 1 bits in a character is even, the C bit is 0.

Standard BCD Interchange Code

The standard BCD interchange code has been developed to provide compatibility among various computer systems. The coding structure consists basically of 64 different characters as shown in Figure 16.2. The collating sequence, graphics, card code, and BCD code for each of the 64 different bit combinations are shown. The C bit, used for parity checking purposes, is dependent upon the specific computer design using the standard BCD interchange code. If the machine is designed for odd parity, a C bit will be automatically placed in each C position of a character that contains an even number of 1 bits. Conversely, a system using even parity will have a C bit placed in each C position of a character that contains an odd number of 1 bits.

Five of the standard BCD bit combinations print out as two different characters or graphics, depending upon the typeset used in the printer. The two variations are called graphic subset 1 and graphic subset 2 as shown in Figure 16.3.

BCD Code	Graphic Subset 1 Print Arrangement A	Graphic Subset 2 Print Arrangement H
8-2-1	#	=
8-4	@	'
A-8-4	%	(
B-A	&	+
B-A-8-4	□	)

Figure 16.3 Graphic subset 1 and graphic subset 2.

Graphic subset 1 is used primarily for computer report writing and most commercial applications, while graphic subset 2 is used for advance programming languages and meets general requirements for mathematical symbolism.

Figure 16.4 shows the preferred standardized terminology for the special characters included in the standard BCD interchange code.

SYMBOL	NAME
⯒	Group Mark
⧧	Record Mark
⧻	Segment Mark
ო	Word Separator
@	At Sign
#	Number Sign
&	Ampersand
+	Plus
*	Asterisk
%	Per Cent
/	Slash
\	Backslash
□	Lozenge
b	Blank
b̸	Substitute Blank
(	Left Parenthesis
)	Right Parenthesis
[	Left Bracket
]	Right Bracket
√	Tape Mark
<	Less than
>	Greater than
=	Equal to
;	Semicolon
:	Colon
•	Period or Point
'	Prime or Apostrophe
—	Minus or Hyphen (Dash)
Δ	Delta

Figure 16.4 Standard terminology for BCD special characters.

Extended Binary Coded Decimal Interchange Code (EBCDIC)

This code uses eight binary positions for each character, plus one position for parity checking. With eight bit positions, 256 different characters can be coded, including uppercase and lowercase alphabetic characters, a much wider range of special characters, and many control characters that are meaningful to input and output devices in a computer system. At present, many bit combinations have no assigned function and are reserved for future use. EBCDIC is the internal coding system of the IBM System/3.

American Standard Code for Information Interchange (ASCII)

This is a seven-bit code developed through the cooperation of the data processing industry communications equipment users. The code was standardized in an attempt to simplify machine-to-machine and system-to-system communication.

The IBM System/3, System/360, and System/370 all have an eight-bit character capacity to correspond with the standard eight-bit byte. Therefore, for use in these systems, the code is expanded to an eight-bit representation, referred to by IBM as ASCII-8. The code may be used for internal processing and input/output purposes in those media for which ASCII has been standardized.

A complete table of these codes is shown in Figure 16.5.

Storage

Computer storage accepts data from input devices, exchanges data with the processor, and retains the completed information to be written by the output devices. Thus, all data to be processed by the system must pass through storage.

Storage is always divided up into locations, positions, or sections, each of which is given a number or *address* so that stored data can be readily located as needed. This arrangement can be thought of as resembling small mailboxes in a post office, each box identified and located by an assigned number. Like a mailbox, a storage location has capacity for a specific amount of information that can be inserted or withdrawn by the processor. To insert or withdraw information in any location, the address must always be known.

Unlike the mailbox, however, whenever data enters computer storage, the previous contents of that location are replaced with the new data. When information is taken *from* a location, the contents remain unaltered. Once put in storage, the same data can be used many times. In effect, an exact reproduction of the information is made available for processing or writing by some output device.

An almost unlimited amount of processing can be carried out before data is returned to storage to be made available as output. In many procedures, all required processing can be completed in one pass through the system. Mechanical movement of the records is therefore greatly reduced in relation to processing. As data is "written" from storage, it is reconverted to punched-card code or to the graphic characters of printed information.

Computer storage normally contains at least three kinds of data:

EBCDIC	Bit Configuration
NUL	0000 0000
SOH	0000 0001
STX	0000 0010
ETX	0000 0011
PF	0000 0100
HT	0000 0101
LC	0000 0110
DEL	0000 0111
	0000 1000
RLF	0000 1001
SMM	0000 1010
VT	0000 1011
FF	0000 1100
CR	0000 1101
SO	0000 1110
SI	0000 1111
DLE	0001 0000
DC1	0001 0001
DC2	0001 0010
TM	0001 0011
RES	0001 0100
NL	0001 0101
BS	0001 0110
IL	0001 0111
CAN	0001 1000
EM	0001 1001
CC	0001 1010
CU1	0001 1011
IFS	0001 1100
IGS	0001 1101
IRS	0001 1110
IUS	0001 1111
DS	0010 0000
SOS	0010 0001
FS	0010 0010
	0010 0011
BYP	0010 0100
LF	0010 0101
ETB	0010 0110
ESC	0010 0111
	0010 1000
	0010 1001
SM	0010 1010
CU2	0010 1011
	0010 1100
ENQ	0010 1101
ACK	0010 1110
BEL	0010 1111
	0011 0000
	0011 0001
SYN	0011 0010
	0011 0011
PN	0011 0100
RS	0011 0101
UC	0011 0110
EOT	0011 0111
	0011 1000
	0011 1001
	0011 1010
CU3	0011 1011
DC4	0011 1100
NAK	0011 1101
	0011 1110
SUB	0011 1111
SP	0100 0000
	0100 0001
	0100 0010
	0100 0011
	0100 0100
	0100 0101
	0100 0110
	0100 0111
	0100 1000
	0100 1001
¢ [	0100 1010
.	0100 1011
<	0100 1100
(	0100 1101
+	0100 1110
|	0100 1111
&	0101 0000
	0101 0001
	0101 0010
	0101 0011
	0101 0100
	0101 0101
	0101 0110
	0101 0111
	0101 1000
	0101 1001
!]	0101 1010
$	0101 1011
*	0101 1100
)	0101 1101
;	0101 1110
¬	0101 1111
–	0110 0000
/	0110 0001
	0110 0010
	0110 0011
	0110 0100
	0110 0101
	0110 0110
	0110 0111
	0110 1000
	0110 1001
7/12	0110 1010
,	0110 1011
%	0110 1100
_	0110 1101
>	0110 1110
?	0110 1111
	0111 0000
	0111 0001
	0111 0010
	0111 0011
	0111 0100
	0111 0101
	0111 0110
	0111 0111
	0111 1000
6/0	0111 1001
:	0111 1010
#	0111 1011
@	0111 1100
'	0111 1101
=	0111 1110
"	0111 1111
	1000 0000
a	1000 0001
b	1000 0010
c	1000 0011
d	1000 0100
e	1000 0101
f	1000 0110
g	1000 0111
h	1000 1000
i	1000 1001
	1000 1010
	1000 1011
	1000 1100
	1000 1101
	1000 1110
	1000 1111
	1001 0000
j	1001 0001
k	1001 0010
l	1001 0011
m	1001 0100
n	1001 0101
o	1001 0110
p	1001 0111
q	1001 1000
r	1001 1001
	1001 1010
	1001 1011
	1001 1100
	1001 1101
	1001 1110
	1001 1111
	1010 0000
—	1010 0001
s	1010 0010
t	1010 0011
u	1010 0100
v	1010 0101
w	1010 0110
x	1010 0111
y	1010 1000
z	1010 1001
	1010 1010
	1010 1011
	1010 1100
	1010 1101
	1010 1110
	1010 1111
	1011 0000
	1011 0001
	1011 0010
	1011 0011
	1011 0100
	1011 0101
	1011 0110
	1011 0111
	1011 1000
	1011 1001
	1011 1010
	1011 1011
	1011 1100
	1011 1101
	1011 1110
	1011 1111
PZ 7/11	1100 0000
A	1100 0001
B	1100 0010
C	1100 0011
D	1100 0100
E	1100 0101
F	1100 0110
G	1100 0111
H	1100 1000
I	1100 1001
	1100 1010
	1100 1011
⑀	1100 1100
	1100 1101
⑂	1100 1110
	1100 1111
MZ 7/13	1101 0000
J	1101 0001
K	1101 0010
L	1101 0011
M	1101 0100
N	1101 0101
O	1101 0110
P	1101 0111
Q	1101 1000
R	1101 1001
	1101 1010
	1101 1011
	1101 1100
	1101 1101
	1101 1110
	1101 1111
RM 5/12	1110 0000
	1110 0001
S	1110 0010
T	1110 0011
U	1110 0100
V	1110 0101
W	1110 0110
X	1110 0111
Y	1110 1000
Z	1110 1001
	1110 1010
	1110 1011
⑁	1110 1100
	1110 1101
	1110 1110
	1110 1111
0	1111 0000
1	1111 0001
2	1111 0010
3	1111 0011
4	1111 0100
5	1111 0101
6	1111 0110
7	1111 0111
8	1111 1000
9	1111 1001
⑄	1111 1010
	1111 1011
	1111 1100
	1111 1101
	1111 1110
EO	1111 1111

Figure 16.5 American Standard Code for Information Interchange (ASCII-8).

1. *Data to be Processed.* These may be elements currently being operated upon, such as arithmetic factors to be used in calculation, record fields, complete records, blocks of records, and so on. As far as storage is concerned, this kind of stored data is *transient*; that is, as soon as one element or piece of data is processed by the central processing unit, the next is moved into storage. Each new data element replaces that which was previously stored. Data is always received from some input device. After manipulation by the processor, the results are transmitted to an output device. The intervening processing may also involve the fetching and storing of intermediate results. This may occur any number of times before the final results are ready to be made available as output.

2. *Constant factors.* These items of data are usually required throughout the carrying out of any given task or procedure: arrays of tables of information, such as pay rates, tax rates, miscellaneous values, reference factors, and so on. These items are stored at some predetermined location. Space may also be allocated for temporary storage of intermediate results. Since the processor usually has a limited number of registers, adders, and accumulators, temporary storage is required as a kind of scratch pad where these results can be referred to as needed.

3. *Instructions.* The instructions to the computer for processing are always placed in storage – at least those instructions under whose control the machine is operating.

Stored Program

All reading, processing, and writing must be done according to an established procedure. Control over the entire system is maintained by placing *instructions* to the machine in storage in exactly the same form as data. Each detailed step of the procedure is coded as an instruction and then placed in known storage locations. These locations must be apart from the transient data or constant factors.

A complete procedure can be developed from individual instructions if they are grouped in proper sequence as a program. Instructions in the program direct the computer to produce desired results. This *stored program* controls all operations of internal processing as well as those of the input and output devices. Once stored, the program can be retained indefinitely during normal system operation. Thus, storage is a type of artificial *memory* that can "remember" instructions and recall them as required. The number of instructions that can be stored, and the consequent length of the program, depends upon the capacity of storage.

When a particular task is finished, new instructions can be *loaded* into storage, replacing the program previously retained. Any of the standard input devices can be used for this purpose because instructions take the same form as data.

The possible variations of the stored program provide a computer with almost unlimited flexibility. A single machine can execute a great many different procedures by simply loading the appropriate program into memory. Most important, because instructions are stored in the same form as data, the machine can operate upon its own instructions. This basic concept is a most remarkable departure from the operating principles of all other types of previously developed calculating or information-handling equipment. It enables the machine to alter its own program in response to conditions encountered while following a procedure. Consequently, a computer can be programmed to exercise a limited degree of judgment, but always within the framework of the operations it can perform and as directed by its human operators.

Central Processing Unit

The central processing unit (CPU) controls and supervises the entire machine system. It performs the actual arithmetic and logical operations, and in a real sense it is "the computer." From a functional viewpoint, the CPU is made up of two sections: control and arithmetic/logical.

The control section has the various facilities required to direct and coordinate all operations called for by instructions in the stored program. These operations include addressing main storage to locate, fetch, and store data or instructions, and initiating communications between storage and the external devices. By means of the control section, automatic, integrated operation of the entire computer system is achieved.

In many ways, the control section can be compared to a telephone exchange. All possible data paths already exist, just as there are connecting lines between all telephones serviced by the exchange.

The telephone exchange has the equipment to carry sound pulses from one phone to another, ring the phones, connect and disconnect circuits, and the like. The path of communication between one telephone and another is set up by appropriate controls and mechanisms in the central exchange. In a computer system, the control section can respond to an interruption from some source, start or stop communication with an input or output device, turn a signal device on or off, fetch or store data and instructions, sequence instructions in the desired order, and so on.

The arithmetic/logical sections contain the necessary circuitry to perform arithmetic and logical operations. The arithmetic section calculates, shifts numbers, sets the algebraic sign of results, rounds, compares, and the like. The logical section carries out the decision-making operations to change the sequence of instructions as may be required. These include comparison, translation, editing, testing indicators, and so forth.

Functional Units

A *register* is another type of storage device capable of receiving some specific unit of information, holding it, and then transferring it as directed by controlling circuitry. Various electronic components may be used, such as transistors, depending upon what function the register is designed to perform. Registers are normally divided into positions or places, each of which can indicate a binary one or zero.

Some types of registers and their functions are:

1. An *accumulator* holds the results of a calculation. When sequences of arithmetic operations are performed, the results in an accumulator are often used from one step in the procedure to the next. For example, two fields may be added to produce a sum in the accumulator. The sum may then be multiplied by another field and the product added to still a fourth field. The final result can remain in the accumulator until transferred to storage for output.

2. A *storage register* contains data taken from, or being sent to, storage. This type of register serves as a place to store data temporarily as it passes between the processing unit and main storage.

3. An *address register* holds the designation of an input or output device or the address of a storage location.

4. An *instruction register* contains the instruction being interpreted or executed.

5. *General registers* may hold several types of data. Register A may contain one factor to be used in a calculation, and Register B the second factor. The two factors can be combined and placed in Register C.

The contents of some registers can be shifted to right or left within the register and in some cases between registers. As the contents are shifted, places vacated can be automatically filled with zeros. The contents of a register can be *masked* in various ways so that individual binary positions can be either selected or ignored.

Registers may temporarily hold information while associated circuits analyze the data. For example, an instruction can be placed in a register and associated circuits can determine what operation is to be performed and locate the data to be used. Data, such as single characters, bytes, or other data segments, may be placed temporarily in registers to check validity of coding or data representation.

In the computer, all data is actually represented with binary indications of ones and zeros.

Usually, the more important registers in a computer system, particularly those involved in normal data flow and storage addressing, have small incandescent or neon lights associated with them. These lights are mounted systematically on an operating or maintenance *console* where they show register contents and various program conditions. With these lights, an operator or maintenance man can "see" into the machine and determine, if necessary, just how instructions are being performed. During this kind of operation, the machine is manually cycled or otherwise "slowed down" so that detailed action of the various registers can be followed. When the system runs at normal speed, the lights usually have no significance but display the blinking effect often shown on news or entertainment media to dramatize the complexity of computer operation.

Console lights are also used in diagnostic and maintenance procedures.

Counters

An adder receives data from two or more sources, performs addition in electronic circuitry and sends the results back to some receiving register. Binary carries to higher order positions are made automatically. The final sum goes to corresponding positions of the receiving register.

Machine Cycles

All operations of the entire computer system take place in fixed intervals of time. These intervals are measured precisely by regular pulses emitted from an electronic clock at extremely high frequencies, perhaps several million pulses per second. Each machine cycle lasts for a fixed number of pulses.

Time references are stated in milliseconds, microseconds, and nanoseconds. The following table lists these terms with their abbreviations and decimal and fractional equivalents.

Decimal Equivalent		*Fractional Equivalent*		*Term*
0.1	=	$\frac{1}{10}$ second	=	100 milliseconds
0.001	=	$\frac{1}{1{,}000}$ second	=	1 millisecond (msec)
0.000001	=	$\frac{1}{1{,}000{,}000}$ second	=	1 microsecond (μsec)
0.000000001	=	$\frac{1}{1{,}000{,}000{,}000}$ second	=	1 nanosecond (nsec)

It is sometimes difficult to understand such extremely short intervals of time unless they are compared with some known reference. For example, "quick as a wink" is actually about one-tenth of a second, or 100 milliseconds. A jet plane traveling at 600 miles per hour covers 10 miles in one minute, 880 feet in one second, 0.88 of one foot in one millisecond and approximately 0.01 inch in one microsecond. Light travels at a speed of 186,282 miles per second, about 982 feet per microsecond, and slightly less than one foot per nanosecond. Or, a nanosecond is about the time it takes the light from this page to reach the reader's eye.

Within the interval of a timed machine cycle, the computer can be programmed to perform a specific operation. The number of individual operations required to execute a single instruction depends upon the instruction. Thus, various machine operations may be combined to execute a single instruction.

Instructions usually consist of at least two parts: an operation and an operand. The operation part tells the machine which basic function to perform: read, write, add, subtract, and so on. The operand can be the address of data in storage. Therefore, a complete add instruction includes both the function of adding and the specification of what data is involved. If the operation is "write," the operand will be the location in storage of the information to be written and the amount of data to be transmitted to the previously selected output device.

Instructions also cause the machine to perform a control function such as shifting a quantity in a register to prepare for division or testing an indicator to determine if a coding error has been detected.

In order to receive, interpret, and execute instructions, the processing unit must operate in a prescribed sequence as determined by the specific instruction and carried out during a fixed interval made up of timed pulses.

Program Execution

Program instructions are always placed in known locations of storage. Each addressable location may contain just one instruction or, depending upon the computer, an instruction may occupy several locations. If instructions are variable in length, a part of the instruction must indicate exactly how many storage positions are occupied. In any case, instructions are stored in exactly the same manner as data.

Instructions are placed in storage in the sequence in which they are normally executed and in ascending locations. That is, the first instruction of the program may be in location 0001, the second in location 0002, the third in 0003, and so on. Or, the first instruction can just as well be placed in location 5421, the second in 5422, and so on, in any convenient space. But in order to be executed, the program must *always* be in storage.

When machine operation begins, an *instruction counter* is set to the address of the first instruction. To begin processing, this instruction is fetched from storage and decoded in a register. Then, while it is being executed, the counter automatically advances or *steps* to the address corresponding to the location of the next stored instruction. In this way the instruction counter acts as a pointer, directing the processor to each instruction as it is required.

If an instruction occupies one position of storage, the counter steps one. If an instruction occupies five positions, the counter steps five. In either case, by the time an instruction is executed, the counter has located the next instruction in sequence. The stepping action of the counter is automatic. The result is that when the computer is directed to any series of instructions in any storage locations, it proceeds to execute them one after another until *instructed* to do otherwise.

For example, assume that a program is stored beginning in location 0142. To start operation, the instruction counter is set to address 0142. The first instruction is fetched from storage and brought to the processor. The operation part of the instruction is decoded. In this way the machine is told what operations to perform: add, subtract, compare, etc. Refer to Figure 16.6.

The operand part is placed in an address register. This tells the machine where in storage to locate the factors to be operated upon. The instruction operand may also indicate where the addition is to be performed, since at least two factors are always involved to produce a sum. The operation of addition is always designed to occur in a specific register or accumulator. The location of the second factor in the addition is therefore predetermined and need not be specified.

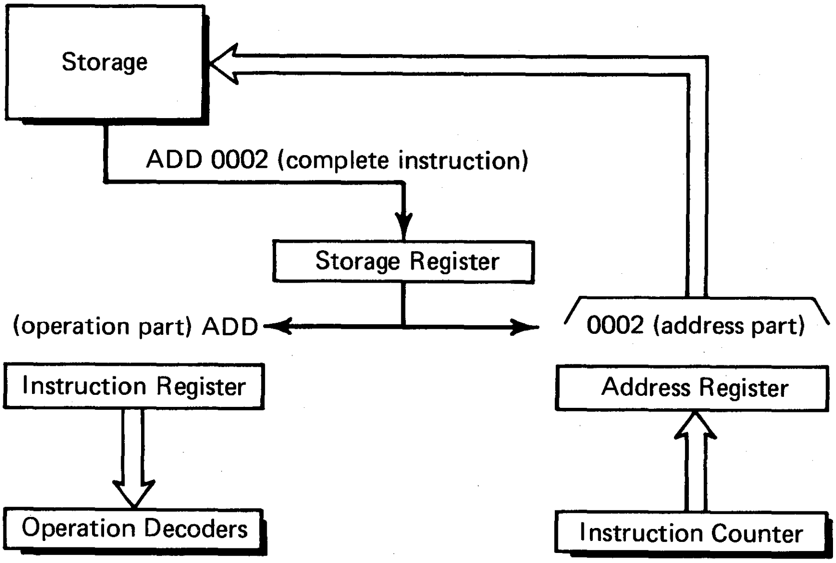


Figure 16.6 Instruction decoding.

The next instruction is located. This instruction may return the sum just developed to storage or fetch and add another factor to the sum. When arithmetic operations are complete, instructions may start an input or output device to send or receive data.

Execution of instructions does not have to proceed sequentially. That is, a control instruction can act directly upon the instruction counter. Such an instruction only directs the machine system; no data is processed. Since the counter is capable of addition, it can be incremented to point to any location of storage by simply adding a constant or variable factor to the address contained by the counter.

The counter can also be reset to the address of the beginning of the program so that all instructions can be repeated for another group of incoming data. Thus, the execution of a program can be repeated any number of times as records are read serially into storage.

The counter can be reset to zero or to any storage address by a *branch* instruction. The program can be arranged so that branching is conditional or unconditional.

For instance, the branch or transfer back to the beginning of the program should occur only if there are more records to process. However, if the input device has fed all records and indicates an end-of-file condition, then the program should branch to a control instruction to terminate or *halt* operation.

The computer can also be directed to examine some indicating device and then branch if the indicator is either on or off. Such an instruction has the effect of saying, "Look at the sign of the quantity in the accumulator. If the sign is minus, take the next instruction from location 5000. If the sign is plus, continue with the next instruction in sequence." The instruction is set in one of two possible location addresses, 5000 or the address of the next instruction. At location 5000, a special series of instructions, or *subroutine*, will direct the computer to carry out special procedures for the handling of minus or credit balances. When that subroutine is completed, a transfer is effected back to the main program.

Thus, the logical path through a program followed by the machine may be controlled either by unconditional branching or by a series of conditional tests applied at various points. However, the actual arrangement of instructions in storage is not normally altered.

Questions and Exercises

1. Explain the structure of the BCD code.
2. Explain the concept of the storage address.
3. Name three kinds of data normally contained in main storage.

17 ORGANIZATION OF A UNIT RECORD COMPUTER: IBM SYSTEM/3 MODEL 10

The elements of any computer system, including the System/3, are each designed to perform specific functions pertaining to data input, processing, data output, and control.

Card Input

The 96-column card serves as the principal input medium for the System/3 Model 10. Two card feeds, primary and secondary, are provided in a multifunction card unit (MFCU). Cards are placed in the hoppers facedown, left end toward the machine. Each feed can be controlled to operate independently of the other. Cards are directed to the read, wait, punch, and print stations by transport mechanisms as shown schematically in Figure 17.1.

When a card is fed from one of the hoppers, it passes first through the read station where light shining through the punched holes in each column is converted to electrical energy by an array of solar cells. There is one solar cell for each punch position. Reading is done serially, column by column, three tiers at a time, from the left end of the card. That is, columns 1, 33, and 65 are read; then columns 2, 34, and 66, and so on. The data in all 96 columns is transmitted to internal storage where it is held for processing. Input data is converted automatically from card code representation to the internal electronic form usable by the processing unit.

4. Can information in storage be used more than once? Why?
5. What are the two main functional units contained in the central processor?
6. Describe the functions of the following: accumulator, storage register, counter, adder.
7. Name the two parts of a computer instruction.
8. Name the following time intervals in seconds: 0.2; 0.004; 0.000001; 0.000000002.
9. Explain the general concept of the stored program.
10. How is a program executed?

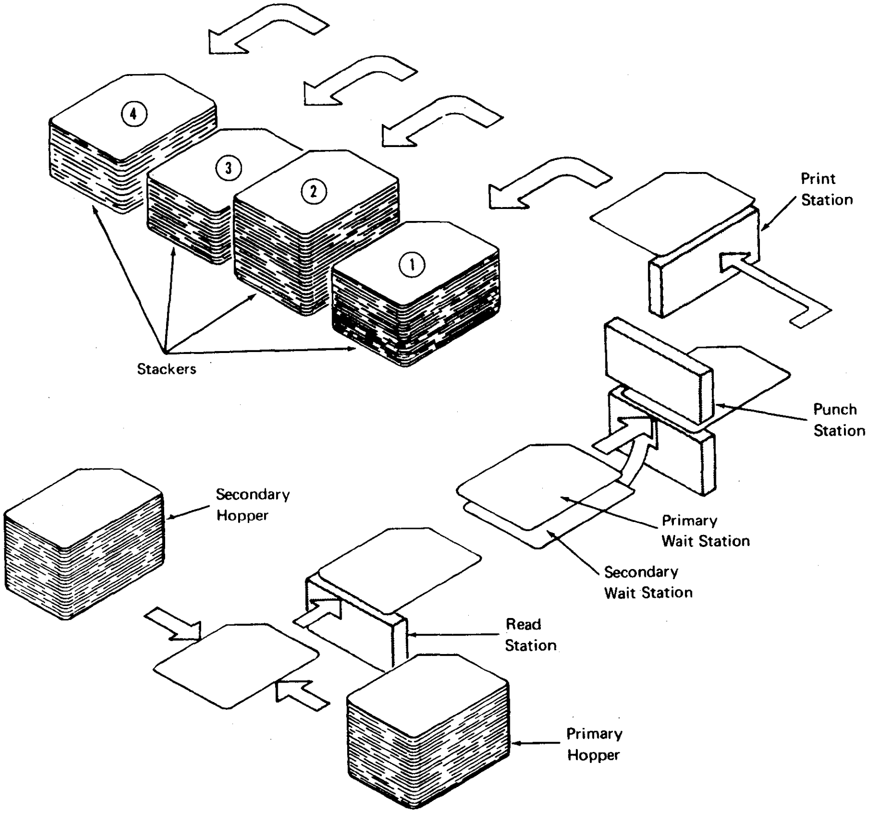


Figure 17.1 Schematic. IBM Multifunction Card Unit.

Input data entering the system is checked to insure reading accuracy. Each set of three tiers is read twice and both readings must agree. Disagreement between the two readings produces a *read check*. The hopper that fed the error card can be determined from indicator lights on the MFCU control panel. The incorrect card can be located at the primary or secondary wait station.

The speed of card feeding and reading is determined by the operations performed. The rated speeds of 250 cards per minute (Model A1) or 500 cards per minute (Model A2) are for reading operations only. If punching or printing is to be performed, the reading rate will be reduced to correspond with the slower of these two operations.

If required, cards may be fed through the reading station without being read. The reading sensors are not actuated and the cards pass through the station without transmitting any data to storage.

Card Output After a card passes the read station of the MFCU, it enters a wait station. Note that there is an upper wait station for the primary feed and a lower wait station for the secondary feed. The card then passes to a punch station where it is punched column by column, three tiers at a time. All punched output data is transmitted from storage and is converted automatically from the internal system representation to punched card code.

Punching each hole in a column generates a signal that represents that hole. After a column has been punched, all the signals for holes in the column are compared with the character code specified for that column. If the two do not match, a *punch check* occurs. Data to be punched from storage is also checked to insure validity within the 64-character set allowed in the card coding structure. An error causes a *punch invalid check.* The card that contains the incorrectly punched data is in the stacker previously selected at the start of the punching operation. Card punching is done at the rate of 60 cards per minute (Model A1) or 120 cards per minute (Model A2). The unit can also be controlled to feed cards through the punch station without punching.

Output cards can be stacked in any of the four stackers. If there is no stacker selection, cards from the primary hopper will be placed in stacker 1; cards from the secondary hopper will be placed in stacker 4.

When collating operations are performed, one file of sequenced cards is placed in the primary hopper and the other in the secondary. Cards can be merged, matched, and selected to simulate the operation of the 80-column collator described earlier.

To either reproduce or gangpunch, blanks are placed in one hopper of the MFCU, original cards in the other. Under control of the processing unit, cards can be reproduced and gangpunched as required by the procedure.

In file maintenance operations, master cards can be fed from one hopper while detail cards are fed from the other. Operations of matching and merging can be combined with calculation, gangpunching or reproducing, and total accumulation.

If summary punching of new balances is required, masters and details are merged first in one operation using the two feed hoppers of the MFCU. In a second operation, the merged files are placed in one hopper and blanks in the other. The MFCU can then be controlled to summary punch into blank cards. In this case, the System/3 can combine the operations of calculating, total accumulation, summary punching, and report printing in one operation.

Printed Card Output

From the punch station, cards pass through a cornering station to a print station equipped with a series of printing wheels. As many as four lines of 32 characters each can be printed on one card. The MFCU prints all of the possible 64 characters in the 96-column card code. Data to be printed is transmitted from storage in the processing unit and may or may not be punched in the card. Cards not printed are ejected through the station to the stackers.

Card throughput, for up to three rows of printing, is 60 or 120 cards per minute. There is a reduction in the rated speed if four lines per card are printed.

Two checks are made on printing accuracy. An error in the synchronization between the print wheels and the MFCU attachment circuitry causes a *print data check.* The check insures that the wheel has been positioned to the proper character as transmitted from storage. An error in synchronization between the MFCU circuitry and a printer stepper clutch causes a *print clutch check.* This check insures the proper positioning of the print lines. The error card is fed to the stacker designated before the start of the printing operation.

Line Printer Output

A line printer is provided with the System/3 to produce reports at high speed as output from the computer. All information to be printed is transmitted by the processing unit from internal storage where each line of data is arranged in a predetermined format. The basic printer can place as many as 96 characters on one line, spaced 10 characters to the inch. The user can obtain an optional feature from IBM which extends the 96-character capacity to either 120 or 132 positions in a single line.

Forty-eight different characters can be printed by the standard machine: 10 numerical, 26 alphabetic, and the 12 special characters . + & $ * - / , % # @ '. No type wheels or bars are used. Instead, multiple arrays of type slugs are placed end to end to form a loop or *chain.* There are two characters on each slug. When the printer is operating, the chain moves in a horizontal plane across the blank paper form as shown in Figure 17.2. The chain is driven

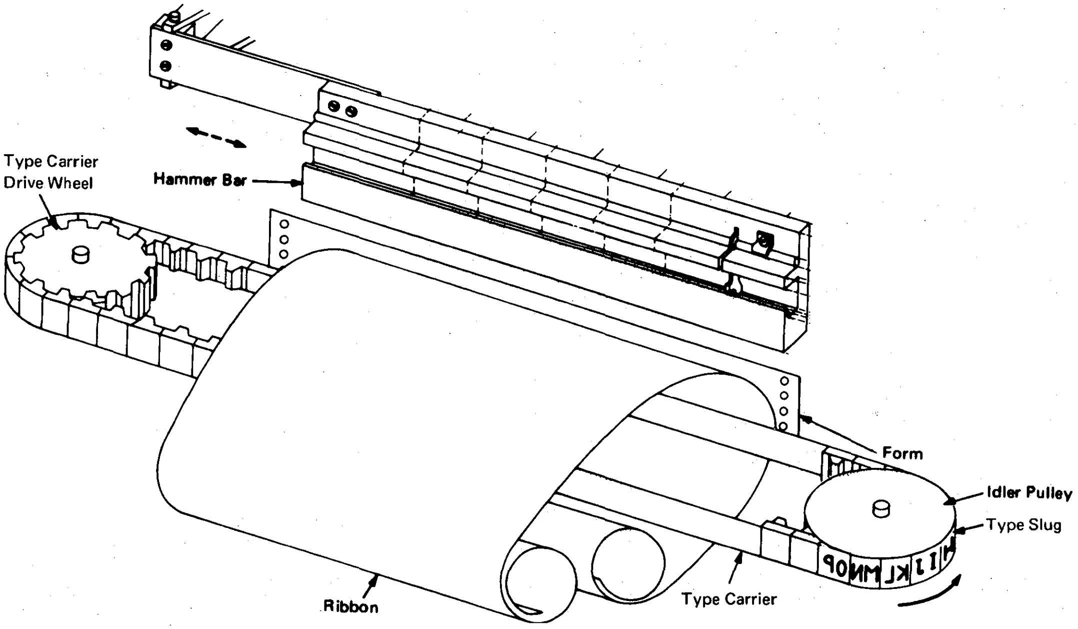


Figure 17.2 Print chain and paper form.

continuously over two sprockets, one on each side of the printing unit. Each chain has a total of 120 slugs or 240 character positions and the standard set of 48 characters is repeated five times. The first array of slugs is depicted below as they appear on the chain.

12	34	56	78	90	#@	/S	TU	VW	XY	Z&	,%	JK	LM	NO	PQ	R-	$*	AB	CD	EF	GH	I+	.’

Printing production is either 100 lines per minute (Model 1) or 200 lines per minute (Model 2).

The usable character set can be extended from 48 to as many as 120 different characters by means of a universal character set feature. For example, the printer can be controlled to print both uppercase and lowercase alphabetic characters. However, when this feature is used, the line production speed is reduced somewhat, depending upon the text being printed.

Printing is accomplished by striking the paper from the back with small hammers, thus forcing the paper forward against the type. An inked ribbon between the paper and the type chain transfers the character impressions to form the printed line. The rotation of the chain and the action of the hammer mechanism is synchronized in such a way that type slugs are struck in the proper positions to exactly duplicate the image of the line in storage. Thus, the arrangement of the line is completely flexible. Unlike punched-card accounting machines, mechanical zero suppression is not required. Printed output does not need to be within a set columnar or tabular format. The chain printer can produce a free-form printed page in exactly the same manner as a typewriter.

The entire print chain cartridge can be removed from the machine by the operator and replaced with another cartridge. This interchangeable type makes it possible to use a number of different character sets or fonts on one machine.

The movement of the paper forms is also controlled from the processing unit. Maximum length of a single sheet is 14 inches. This is 112 lines at eight lines per inch, and 84 lines at six lines per inch. The 6- or 8-line vertical spacing can be selected by the operator. Spacing between lines and skipping to a predetermined line on a page are controlled from the processing unit.

As another optional feature, a dual-feed carriage enables the printer to produce two side-by-side documents at the same time. Each document can be independently controlled.

A number of checks are made automatically to insure printing accuracy, including those on the action and synchronization of the printing hammers and chain. A print check error is indicated when the hammer circuitry does not respond properly to a print signal. A thermal check is made to prevent the hammer unit from overheating. Other indicators show a forms jam, carriage slipping or spacing too far, or various interlock conditions in the chain and forms chute.

An indicator also shows if a character that is to be printed is not available on the chain. This checks that the proper type cartridge is on the machine.

Processing Unit

The processing unit of the System/3 is divided into three main parts:

Storage
Control
Arithmetic/logical

Storage

The basic storage size for the system is 8,192 eight-bit words or bytes. Additional storage capacity can be obtained in increments of 4,096 bytes up to a total of 32,768 bytes. Storage capacity for computers is often referred to in increments of "K" meaning 2^{10} or 1,024. Therefore, basic System/3 storage is 8K (*8 X 1024 = 8,192*) bytes with additional sizes available as 12K, 16K, 24K, and 32K.

In addition to the eight data bits, each byte contains one parity bit to maintain an odd number of one bits per byte. Each time a byte of data is used, it is checked to insure that it contains an odd number of one bits, including the parity bit. If an even number of ones is detected, the processing unit indicates a process check.

Although the parity bit is really an inherent part of the internal coding system, the bit is "invisible" to the programmer. It represents no data and cannot be manipulated in any way by instructions. Capacities are stated and thought of in terms of eight-bit data representation, but it is understood that the byte always includes the ninth parity bit when used within the machine system.

Data is stored in extended binary coded decimal interchange code (EBCDIC). Numeric data can also be handled as *zoned decimal format* where each byte is considered to be divided into two groups of four bits each. Bits 0 through 3 make up the zone portion; bits 4 through 7 make up the digit portion as shown in Figure 17.3. Whenever data is handled in this format, the zone bits do not participate in any arithmetic operations.

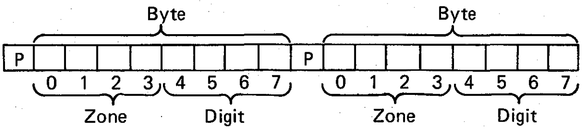


Figure 17.3 Zoned decimal format of data storage.

A group of bytes may form a field in adjacent locations of storage just as adjacent card columns may form a punched card field. When the field is numeric and in zoned decimal format, the zone bits of the low-order byte indicate the sign of the field: 1111 for plus, and 1101 for minus.

Data can also be handled in binary format as an eight-bit binary integer as shown in Figure 17.4. Note that all data in storage is in bytes of eight binary bits. The format of instructions executed by the processing unit determines whether the data is treated as zoned decimal, graphic characters, or binary integers.

Each of the 96 columns of input card records can contain one six-bit character. During input operations, card data is converted automatically to EBCDIC. During output operations, EBCDIC is converted back to card code or to printed graphics.

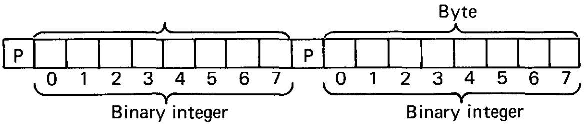


Figure 17.4 Binary integer format of data storage.

Addresses of storage locations are consecutively numbered in hexadecimal from 0000 to the upper limits of the largest size storage. Any field or group of bytes can be located by addressing either the leftmost or rightmost (lowest or highest address) byte, depending upon the instruction.

Processing Unit Data Flow

Figure 17.5 is a simplified schematic of the System/3 processing unit data flow. Reference numbers on the illustration correspond to the item numbers below, as follows:

1. Information enters the processing unit by way of the *channel* and is translated automatically to EBCDIC. A parity check is made, byte by byte, on the code translation.

2. Incoming data is routed to *storage* by the control section and placed in the locations designated by an I/O instruction.

3. The *storage data register* stores data temporarily as it passes between processing and storage.

4. The *A and B registers* serve as temporary storage for data to be processed by the arithmetic/logical unit. Each of these registers holds one byte of data and each can be entered from several other units in the data flow.

5. The *storage address register* holds the address to be accessed in storage.

6. The *condition register* stores one byte. Each bit indicates the result of specific processing operations. For example, bits are placed in the condition register to indicate a high, low, or equal

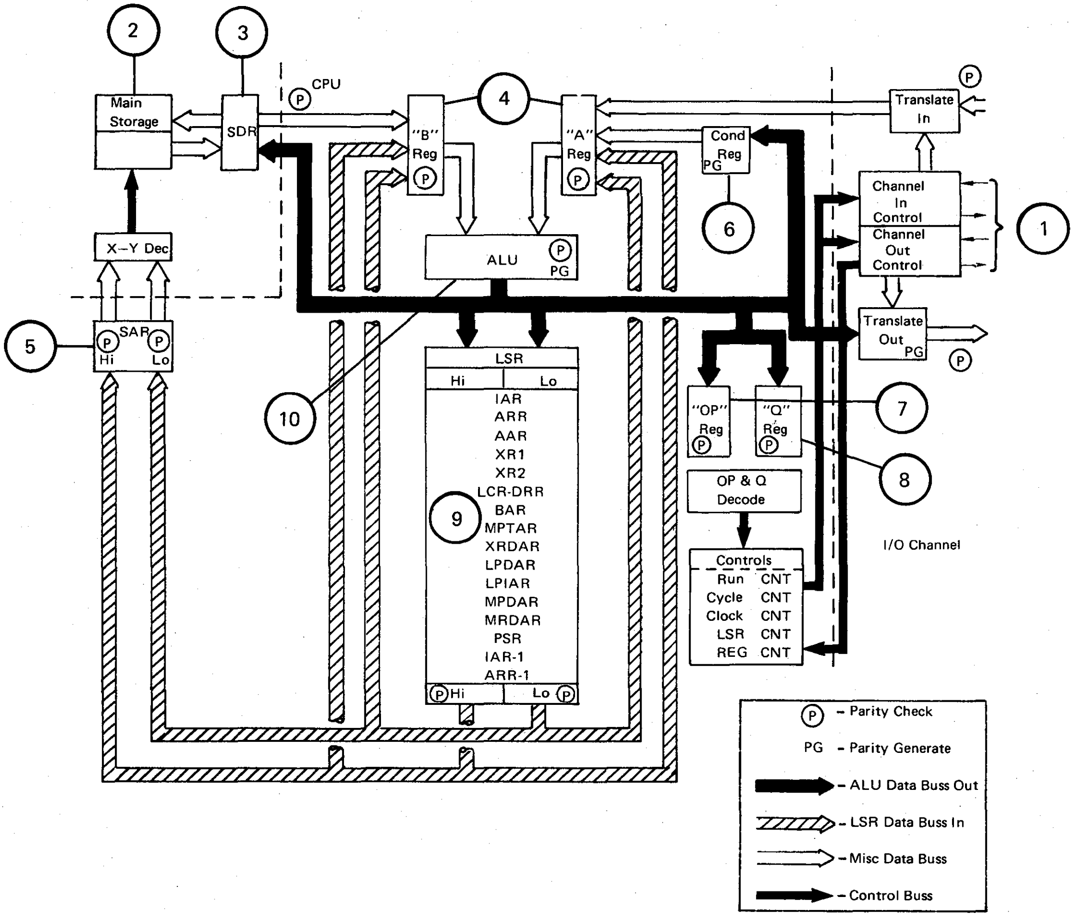


Figure 17.5 Schematic. System/3 Model 10 Data Flow.

comparison. After an arithmetic operation, bits may indicate that a binary or decimal overflow has occurred. The assignment of each register position to indicate a condition is as follows:

Bit	*Name*
7	Equal
6	Low
5	High
4	Decimal overflow
3	Test false
2	Binary overflow
1	Unassigned
0	Unassigned

Instructions can test the register for these conditions and direct the machine to execute corresponding program routines accordingly.

7. The *op code register* holds the operation code byte of each instruction taken from storage. When an operation is decoded, the control circuitry is actuated to perform that specific function. For example, if the operation code is hexadecimal 06 (0000 0110), the arithmetic/logical unit is directed to add two fields and place the sum in storage. If the operation code is hexadecimal 07 (0000 0111), the unit is directed to subtract one field from another and place the remainder in storage. The addresses of the fields are designated in the operand part of the instructions.

8. The *Q register* holds a special Q byte from the instruction that is to be executed. The byte is read into circuitry to control the operations performed by instructions. For example, in the operations of add and subtract zoned decimal, the Q byte designates the length of the fields that are to be operated upon.

9. There are 16 *local storage registers* in the basic card system that also contain data required for the execution of various instructions. The function of these registers will not be explained here.

10. The *arithmetic/logical unit* performs all arithmetic and logical functions on the processing unit. It is capable of decimal add and subtract, binary add and subtract, compare, and other data manipulating functions.

Note that at various points in the data flow, parity checks are taken as the data moves through the processing unit. Also, as new data is generated, parity is generated at the same time. That is, when a sum or remainder is returned to storage, the parity portion of the byte is generated and returned also.

Each operation performed is done in two phases: *instruction* and *execution.* Some instructions combine the phases so that there is no distinct execution phase.

The time interval in which the processing unit reads one byte from storage is known as a *cycle.* At least three cycles must be performed for every instruction. Cycles can also be taken by the input and output units in sending and receiving data.

The reservation of storage areas for either data or instructions is originally done by the programmer. Actually, the processing unit cannot distinguish between data and instructions. If by mistake the instruction counter is pointed to data, the machine will erroneously attempt to operate according to whatever information is placed in the operation code register. Improper functioning of the computer would undoubtedly result and quickly halt the system.

Conversely, however, instructions can legitimately be treated as data. For example, in item 7 above, the operation code for add zoned decimal was given as hexadecimal 06, while the code for subtract was given as hexadecimal 07. Therefore, by simply adding a binary 1 to the code for addition, the operation is changed to subtract. Or, by taking 1 from the subtract code 07, the remainder becomes the operation code for add. In a program, instructions can be given to test the condition register for the sign of a result. If the sign is plus, the machine may be instructed to subtract a given factor to obtain some final result. Or, if the sign is minus, the machine may be instructed to *add* the factor to reach the final result. That is, the program can instruct the computer to alter other instructions to either add or subtract a factor, depending upon the sign of the previous result.

The processing unit also acts as a controller for all I/O devices. The unit communicates with the devices by way of an interface called the *input/output channel.* Each device attached to this channel has an address in the same manner that each storage location in the processor has an address. The device can thus be identified and activated by instructions in the stored program. The I/O channel consists of:

1. A set of signal lines that carry information to and from the processing unit;

2. Logic to translate card code into EBCDIC and vice versa;

3. Logic to *time share* I/O operations with processing and to *interrupt* the main program for I/O servicing, as required.

Time Sharing

In this mode, internal data handling can be carried out while input or output information is being routed between an I/O device and main storage. The I/O device is directed to start operations by a specific instruction and is then left to perform its operation until storage is required. The device then *steals* one or more cycles from other processing unit operations in order to store or retrieve data. The processing unit then continues to perform other operations until the I/O device again requires storage cycles. This *interleaving* of functions is possible because the processor can move data into storage in a small fraction of the time required to complete an I/O cycle, such as feed a card or print a line.

An optional *dual programming* feature provides the system with the capability to execute two independent programs on a time-sharing basis. The feature allows the processing unit to transfer to a different program when the current program must wait for completion of an I/O operation. This uses the high performance capabilities of the processing unit rather than forcing it to wait for completion of the execution by active I/O devices. The feature allows two independent programs to be placed in storage at the same time.

Interrupt

Normally, the *interrupt* capability of computers implies that the processing unit can interrupt a currently operating sequence of instructions to perform some other instruction sequence required by an I/O device, and then return to the main or interrupted program. For example, certain I/O units require special program subroutines to handle data to be entered within a limited period of time. Such a situation can occur when a communication device is ready to send messages to the computer and may require servicing before a current sequence of instructions has been completed. The System/3 processor is equipped with an interrupt capability to service an input or output unit whenever the device has information to transmit or is ready to receive.

When interrupted, the system is so designed that:

1. Information is saved to enable the machine to restore itself to the same status it had before the interruption occurred. This may include the status of the condition register, storage registers, and so on.

2. Information is saved to enable the processor to return to the program at the last instruction to be executed from the point where the interruption occurred. This means that after completing the subroutine, the instruction counter must be reset to the address of the next instruction in the main program.

Instruction Format

The System/3 is designed to perform 28 different operations including decimal addition and subtraction, binary addition and subtraction, moving and editing data, testing and changing the contents of registers, controlling the I/O devices, comparing, branching, and others. The functions of these operations are complementary in such a way that the system can meet nearly all data handling requirements of a general-purpose commercial machine. Each possible operation that can be performed is coded as an eight-bit byte.

It is interesting to note that the arithmetic operations of division and multiplication are not included in the operation set. However, by programmed subroutines, the machine can divide and multiply just as though this calculating ability were built into the circuitry. It is a common characteristic of computers that the operation set is generally designed to include only basic functions while more specific or specialized functions must be simulated by programming.

In the System/3, the operation code is combined with a Q byte and an operand to form an instruction. The operation code of the instruction always tells the processor what to do; for example, add zoned decimal, test, branch, etc. The operand of the instruction normally tells the processor where in storage to find the data upon which to operate. Therefore, an operand is usually one or more storage addresses. The Q byte serves to further define the instruction by various means, such as indicating the number of bytes in the operand, the address of an I/O device, registers to be used in the operation, and so on.

Figure 17.6a shows the format of an instruction that will clear a designated area of storage (Operand 1) and replace it with a specified zoned decimal field obtained from another area of storage (Operand 2). The execution of the instruction has the same effect as a clear and add operation in a desk calculator or adding machine. In the System/3, the operation is called *Zero and Add Zoned.* The instruction can be used as the first step in preparation for accumulating quantities in some record field in storage.

Since two storage areas are involved, the instruction contains two operand addresses and each address is two bytes in length. The address of Operand 1 is shown as hexadecimal 10, the location of the units position of the numeric field. The address of Operand 2 is shown as hex 20, also the location of the units position. The two operands as they appear in storage are shown in Figure 17.6b, c.

The Q byte in this instruction format designates the length of the two operands involved. Positions 4 through 7 of the byte designate the number of bytes in Operand 2, minus one. Since Operand 2 is actually three bytes long, positions 4 through 7 contain a hex 2. Positions 0 through 3 of the Q byte designate the number of bytes by which the length of Operand 1 *exceeds* the length of Operand 2. Since Operand 2 is five bytes in length, positions 0 through 3 also contain the quantity 2. For this operation in the System/3, the maximum length of Operand 1 is 31 bytes and the maximum length of Operand 2 is 16 bytes. These restrictions are imposed by the design of the system.

Instruction

(a)

04	22	00	10	00	20
Op code	Q Byte	Operand 1		Operand 2	

Operand 1 before instruction execution

(b)

F7	F6	F3	F6	F9	Data
000C	000D	000E	000F	0010	Address

Operand 2 before instruction execution

(c)

F4	F2	F5	Data
001E	001F	0020	Address

Operand 1 after instruction execution

(d)

F0	F0	F4	F2	F5	Data
000C	000D	000E	000F	0010	Address

Figure 17.6 System/3 instruction format. Zero and add zoned.

The operation code for Zero and Add Zoned is coded as 04 in hexadecimal, indicating that the eight-bit byte in storage would be indicated as 0000 0100.

When the instruction is executed by the processor, Operand 2 is placed byte by byte into the storage locations of Operand 1. Extra high-order zeros are inserted into those positions by which Operand 1 exceeds Operand 2 in length. The zone bits in each byte of the result are all set to ones, except the units position. This byte carries the sign of the result field: 1111 if positive, 1101 if negative. A result of zero is considered positive.

The result as it would appear in storage is shown in Figure 17.6d. Note that all zoned decimal digits appear with binary ones in the zone portion of the byte. Therefore the hex symbols for the result field are F0, F0, F4, F2, and F5.

Operand 2 remains in storage unchanged (unless the two fields overlap), and can be used in any other data handling operations that may be executed later on in the program. The condition register is set as follows:

Bit	*Name*	*Setting*
7	Equal	Result is zero
6	Low	Result is negative
5	High	Result is positive

Bits 4 through 0 are not affected. In this example, the five-bit position of the register is set to 1. When a result is zero both the 5- and the 7-bit positions are set to 1.

To continue accumulating in the field designated as Operand 1, an operation called *Add Zoned Decimal* can be performed. The instruction is the same as that of the previous example, as shown in Figure 17.7a. Operand 1 remains in the same storage position with an address of hex 0010. The address of the new Operand 2 is hex 0020. Refer to Figure 17.7b, c.

The operation code for Add Zoned Decimal is hex 06. The Q byte remains the same since Operand 1 is five bytes in length and Operand 2 is three bytes. The maximum number of bytes, and consequently the size of the fields that can be added together, also remains the same.

When this instruction is executed, Operand 2 is added algebraically to Operand 1, byte by byte, and the result is stored in the location of Operand 1. The operand fields are addressed by the units position bytes. The zone bits in each byte of the result are all set to ones, except the units position. This byte carries the sign of the result field: 1111 if positive, 1101 if negative. A zero result is considered positive. The result of the addition is shown in

Figure 17.7d as it would appear in storage. Operand 2 remains in storage unchanged (unless the fields overlap). The condition register is set as follows:

Bit	*Name*	*Setting*
7	Equal	Result is zero
6	Low	Result is negative
5	High	Result is positive
4	Decimal overflow	Carry occurred from the high-order position of Operand 1

Bits 3 through 0 are not affected. In this example, the 5-bit position of the register is set to 1.

Instruction

(a)

06	22	00	10	00	30
Op code	Q Byte	Operand 1		Operand 2	

Operand 1 before instruction execution

(b)

F0	F0	F4	F2	F5	Data
000C	000D	000E	000F	0010	Address

Operand 2 before instruction execution

(c)

F2	F6	F3	Data
002E	002F	0030	Address

Operand 1 after instruction execution

(d)

F0	F0	F6	F8	F8	Data
000C	000D	000E	000F	0010	Address

Figure 17.7 System/3 instruction format. Add zoned decimal.

To carry out these arithmetic operations, any area of storage can be used as an accumulator for decimal quantities. An operation of *Subtract Zoned Decimal* (hex code 07) can also be performed to arrive at a net balance in Operand 1.

The following formula is given to calculate the time required by the Sytems/3 to execute the two instructions described above:

$$\textit{Time in } \mu sec = 1.52\ (N + L1 + L2) + 1.52\ (R)$$

where N = instruction length in bytes
$L1$ = length of destination field in bytes (destination field is that field addressed by Operand 1)
$L2$ = length of source field in bytes (source field is that field addressed by Operand 2)
R = length of Operand 1 in bytes when recomplementing is necessary (not required in above examples)

Substituting in the formula:

$$\begin{aligned} \textit{Time} &= 1.52\ (6 + 5 + 3) + 1.52\ (O) \\ &= 21.28\ \mu sec \text{ or, for the two instructions} \\ &= 42.56\ \mu sec \end{aligned}$$

Printer-Keyboard

The System/3 Model 10 can be equipped with a printer-keyboard which is intended to serve primarily as an inquiry station and secondary printer. However, the device can also be used for direct data entry into the system and for communication between operator and machine. The unit is conveniently mounted on the console work table with a forms stand on the floor behind it. Although similar in appearance to an electric typewriter, the keyboard and printer operate independently of each other under stored program control.

The *keyboard* will accept data whenever a "proceed" light is on. Key depressions cause the corresponding character to be transmitted to processor storage on an interrupt basis. However, because the keyboard and printer are not physically linked, key depressions do not automatically cause a character to be printed.

There are 44 character keys as well as shift, space, and carriage return keys, all arranged in typewriter style (Figure 17.8). Functional keys and indicator lights are located to the right of the keyboard for the operator's use.

The *printer* operates at a rated speed of 1.5 characters per second. Printing is serial, character by character, across a $12\frac{1}{2}$-inch writing line at 10 characters per inch. All printing is transmitted from processor storage.

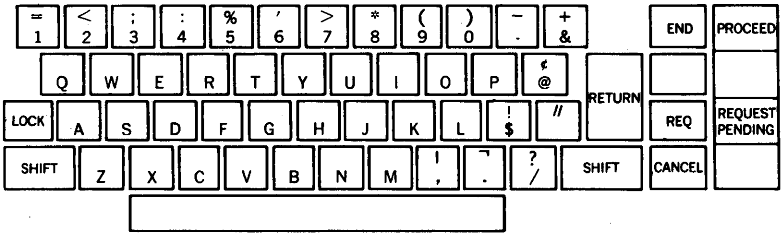


Figure 17.8 System/3 keyboard character arrangement.

Data Entry Keyboard

This unit is used with the processing unit and the MFCU for on-line data recording and data verification. Mounted on the console work table, the keyboard operates under stored program control.

The keyboard has the touch and format of the 5496 Data Recorder, making operator transition from one device to the other quite convenient. The 64-character set is equivalent to that of the 5496. Card punching, reading, and column-for-column printing is provided by the MFCU. Completed cards are placed in the fourth stacker of the MFCU where they are accessible to the operator from a seated position at the keyboard.

The 96-column cards can be punched and verified at the keyboard in exactly the same manner as carried out with a 5496.

80-Column Card Compatibility

New users of the System/3 often change from existing 80-column punched-card equipment. In these cases there is a requirement to convert existing 80-column card files to 96-column cards, particularly master files and other permanent records. To meet this need, IBM makes available an 80-column card read punch for attachment to the System/3. Not standard equipment, the device is normally used on an IBM system that may be rented on an hourly basis in a Basic System Center. Once the files are converted, the user usually does not require regular 80-column input in his application. The one-time conversion of files can be done in advance of actual installation of the new system.

The 80-column files are read by a special card reader and are punched into 96-column cards by the MFCU. The operation is under control of a stored program available from IBM.

Questions and Exercises

1. Name the four card stations of the MFCU. What function is performed at each station.
2. How is the MFCU similar to the 88 Collator in card handling?
3. How is the MFCU similar in function to the 519 Document-Originating Machine?
4. How are files arranged in the MFCU for summary punching?
5. What takes the place of a control panel in the MFCU and the printer?
6. What purpose does the printing function of the MFCU serve?
7. List some advantages of the printing capability of the System/3 line printer compared to a punched-card accounting machine.
8. How are paper forms positioned and controlled on the printer?
9. How are the printer and carriage controlled?
10. How does the processor distinguish between instructions and data?
11. What is an I/O channel?
12. Why is the "interrupt" capability important?
13. What are the three main parts of the instructions presented in this chapter and what is the function of each?

18 IBM SYSTEM/3 MODEL 10: CARD- AND DISC-ORIENTED SYSTEMS

Using the configuration of equipment described thus far, the System/3 Model 10 can perform the data processing operations previously done by single-purpose unit-record machines. A number of these operations can often be combined in one run of the cards through the machine.

Card-Oriented Programs

The System/3 is controlled for each type of procedure by a stored program of instructions in processor storage. The programs are of two general types: utility-oriented and application-oriented.

Utility programs are offered as products by IBM and other vendors from whom they may be obtained by computer users. The following utility programs are typical:

Reproduce/Interpret Program

This program has a number of options. Information from one set of 96-column cards is reproduced into a duplicate deck. Reproducing is on a column-for-column basis. Punched cards can also be printed by the MFCU.

Information from one card deck can be punched into different field positions of blank cards. The user punches format cards to load with the program to specify the arrangement of fields in the duplicate deck. The printing location corresponds to the location of punching in the reproduced deck.

96-Column List Program

This program lists cards on the printer without reformatting. The user may select single-, double-, or triple-spacing; automatic overflow is provided on the standard 11-inch form. After listing has been completed, a card count is printed. The user can also select an option to bypass the listing and obtain only a card count as output.

Sort/Collate Program

This program is designed to perform a wide variety of sorter and collator functions. A specification sheet is used to describe the job to be performed. Cards are punched from the sheet and are used by the program to perform the desired functions. The following are available: sort, merge, match, select, and sequence check.

Selection may be accomplished based upon a variety of logical tests including comparison of field contents to a constant or to other field contents. Several tests may be combined for easy separation of cards which meet certain tests. Tests for zones, digits, or characters may be used during a run to select only those cards to be processed, leaving the remainder of the deck in its original sequence. Specific cards may be directed to a pocket of the MFCU during a match or select run.

Control fields for the various functions may be in ascending or descending sequence and in combinations with numeric fields. Multiple and split control fields may be described, provided they do not exceed an accumulated total length of 100 characters.

In collating operations, the user may substitute a value table which changes the normal sequence values of the sort field contents. It is also possible in the sort program to assign new values to existing punches in control fields for purposes of sequencing, sorting, and matching. Thus, an A may be recoded to a B so that As and Bs would be considered equal. Similarly, the values of A and B can be reversed to provide a new sort pattern.

The control field for sorting or collating need not be in the same location in all cards in the deck. For example, the man number might be in two different locations in two different types of cards being sorted together.

Note that on-line computer sorting differs considerably from sorting practice off-line with special purpose equipment. With the System/3, the sort is begun by placing approximately half of the card file in the MFCU primary feed and half in the secondary feed. The machine attempts to merge the two groups of cards according to a given control field. In-sequence groups or *strings* of records are built up in each of the output stackers. After the initial merge, the machine is programmed to stop and print out for the operator the number of additional passes that will be required. Cards from the output stackers are returned to the feed hoppers: stackers 1 and 2 to the primary feed, and stackers 3 and 4 to the secondary feed. The process is repeated until the entire file is merged into one long string and is therefore in sequence.

Figure 18.1 is an example of this type of sorting. Assume there are 10 records in random order that are to be sorted in ascending sequence by a two-digit number. The deck is divided into two groups and merged in the MFCU. The first two cards (62 and 97) are placed in ascending order in stacker 1. At this point, there is a step-down in sequence because 06 and 18 are both lower in sequence than the last card placed in stacker 1. Therefore, a new string is started in *stacker 3*. Merging continues until the next step-down occurs. When it does, the next string is started in stacker 2. A fourth string is required and the last two cards are placed in stacker 4. When additional strings are produced when sorting larger decks, they also fall into stacker 1, 3, 2, and 4, in that order.

For the second pass, cards from stackers 1 and 2 are placed in the primary feed; cards from stackers 3 and 4 are placed in the secondary feed. Two strings result, and each is placed in stackers 1 and 3 respectively. When the last two strings are merged in the last pass, all cards are in sequence in stacker 1.

Unlike off-line sorting where records are sequenced one column at a time, the computer reads the entire control field at once. Therefore, the total time required to sort depends upon the number of cards being sorted, the degree of presequencing, and the speed of the machine. The shortest possible sorting time results when the records are already in sequence and only one merge pass is required. The longest time for sorting occurs when the deck is in complete reverse order. Normally, when cards are in random order the number of strings may be assumed to equal one-half the number of cards. For a file of 5,000 cards the sorting time can be estimated as follows:

1. Let S equal one-half the number of cards in the file. Example: $S = 2{,}500$.

2. Find the highest power of 2 that is less than S. This number is the probable number of merge passes required. Example: $2^{11} = 2{,}048$ or 11 passes will probably be required.

3. Time in minutes for sorting is $T = NC/R$

where N = number of passes
C = number of cards in file
R = read speed of MFCU.

$$\text{Example: } T = \frac{11 \times 5{,}000}{250} \quad \text{or} \quad \frac{11 \times 5{,}000}{500}$$

T = 220 minutes with MFCU Model A1 or 110 minutes with Model A2.

Data Recording Program

A System/3 equipped with the optional data entry keyboard can be used for on-line data recording with 96-column cards as output. The MFCU is used as a punching device; the keyboard is used for entering the data from an original document. The program controls the system to perform most of the functions of punching and interpreting of the 5496 Data Recorder. Key depressions cause corresponding characters to be entered into processor storage. When the stored record is complete, a card is punched out by the MFCU. There are two advantages:

1. In a low-volume application, the computer can be used to perform all the operations of data recording without the additional cost of providing a separate machine unit.

2. In those installations that have one or more data recorders, the computer can be used to extend card punching capability to take care of peak loads of short duration.

Data Verifying Program

The System/3 may be used to verify the accuracy of recorded data in 96-column cards. The cards may have been previously

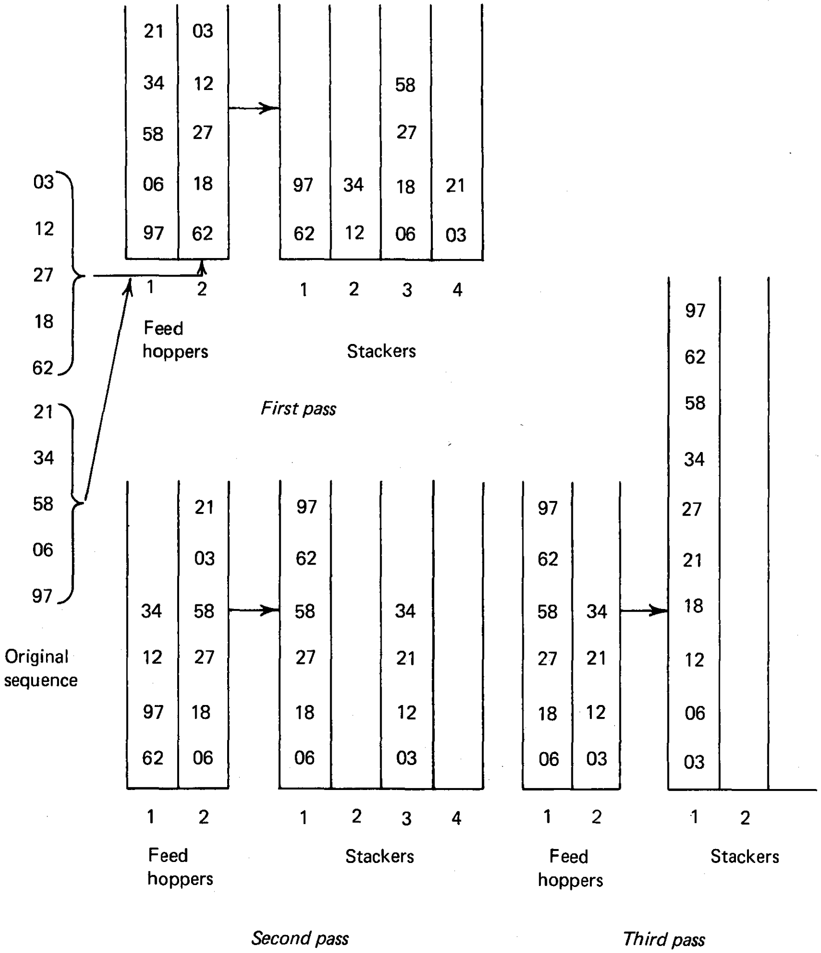


Figure 18.1 Forming sequences or "strings" by sorting on MFCU.

punched by the MFCU or by the 5496 Data Recorder. The MFCU is used in this operation to read the previously punched cards into storage. Blank cards may also be fed from the second hopper for correction of errors.

As each character is keyed from the original document, it enters processor storage where it is compared against the corresponding character in the card record read by the MFCU. If a comparison is not identical, an error light is turned on and the operator may key the original character again. If there was a keying error, the comparison should now be correct and verification can continue. If the card character is incorrect in storage, the error light remains on.

After the third re-try, the keyed character is assumed to be correct and is transmitted to storage. The balance of the record is then keyed. When verification is completed, the correct record in storage may be punched into a blank card and automatically inserted into the card file to replace the error card. The error card will be stacker selected.

A verify OK notation is printed on the fourth print line of each correct card.

80- to 96-Column Conversion Program

This program controls a System/3 equipped with a special IBM Card Read Punch and associated attachments to read 80-column cards, translate the punched data, and punch out 96-column cards in the MFCU. A number of options are provided by the program, including standard character-for-character translation, reformatting the data in the 96-column card, special translation for special characters, and ability to handle invalid and unwanted characters.

The program does not handle conversion from 96 to 80 columns. If this operation is to be done, the user must prepare his own program which can be used only with a System/3 equipped with the special card read punch.

Disk-Oriented System

The System/3 Model 10 can be equipped with disk storage to provide the computer with the advantages of direct access to large amounts of data on line. Two IBM units are available: the 5444 and the 5445 Disk Storage Drives. As many as 7,370,000 eight-bit bytes can be made accessible with two 5444 drives. Nearly 41,000,000 eight-bit bytes can be made accessible by attaching two 5445 drives to the system.

The following section discusses basic operating principles that apply to many direct-access storage devices. Operation of the IBM units is described for purposes of illustration.

Disk Storage Operation

Flat metal disks, similar in appearance to phonograph records, serve as storage media when mounted on disk storage drives. The disks are coated on both sides with a magnetic recording material.

Depending upon the type and model, one or more disks are rotated at a constant speed on a common spindle. For example, disks on the 5444 rotate at the rate of 1,500 revolutions per minute.

One or more disks can be removed from the drive units in a *disk cartridge* or *disk pack*. These removable disks are interchangeable among specified drives in much the same way that phonograph records are interchangeable among record players operating at a given speed and equipped with a proper pickup arm. A 5440 Disk Cartridge, containing one disk, can be used with any 5444 drive and consequently is accessible to any System/3 with a 5444 attached. The 5445 drives are equipped with a 2316 Disk Pack, each containing 11 disks. By following System/3 programming conventions on an IBM System/360 or System/370, data on a 2316 pack can also be accessed by any of these systems. In this way, disks provide virtually unlimited storage capacity that can be accessed by one or more types of computer systems.

Unlike phonograph records, however, information can be recorded as well as written on the disks by any controlling system. Thus, a file produced by one computer can be changed and updated by another.

All data is written in concentric *tracks* on the disk surface by movable read/write heads. These heads are attached by support arms to a carriage that slides back and forth across the disk surfaces. There is one head for each surface, as shown schematically in Figure 18.2. When the drive is running, the heads float on a

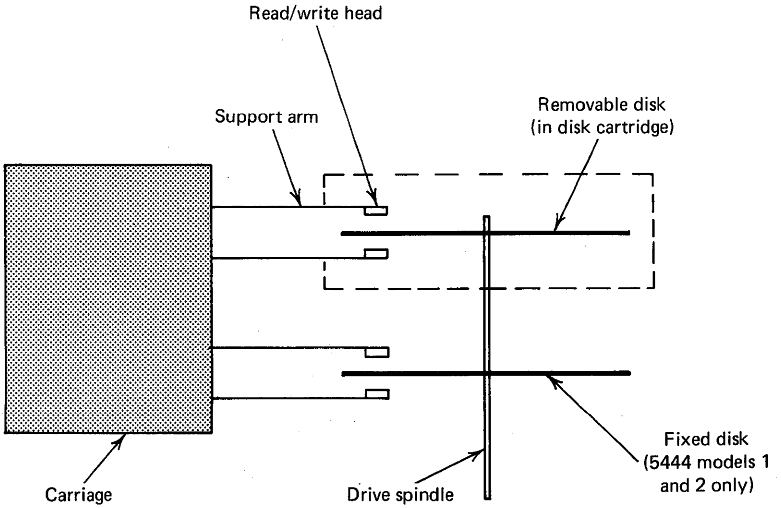

Read/write head
Support arm
Removable disk (in disk cartridge)
Fixed disk (5444 models 1 and 2 only)
Carriage
Drive spindle

Figure 18.2 Schematic. Disk storage drive operation.

thin film of air just clear of the surface of the rotating disks. The heads do not actually touch the surfaces.

Disk drives are provided with contamination control devices to avoid interference between the read/write heads and the disk surfaces. Dust and dirt particles must be excluded from the drive and from the disk cartridge or pack. For example, the 5444 is equipped with a brush that sweeps dust particles from each disk surface during the starting sequence when the drive is first used. The disks run in a closed air circulation system, so that no air enters or leaves. The cartridge is specially designed to prevent off-line contamination when the disk is not on the drive.

The 5444 contains safety devices to control both the start and stop procedures and the various machine operations. These devices include interlocks to prevent the operator from gaining access to the drive while it is operating and also to prevent the drive from starting while the operator has access to it.

The 5444 also has circuits to protect recorded data. These circuits insure that read and write operations take place only when it is intended for them to do so, and that recorded information is not lost accidentally.

Disk Storage Organization

All data on disks is magnetically recorded along concentric circular paths or *tracks*. Each track is placed in a specific position where it can be located by the access mechanism and the attached read/write heads. Each disk on the 5444 can contain either 100 or 200 tracks, depending upon the model. A schematic of the track arrangement is shown in Figure 18.3.

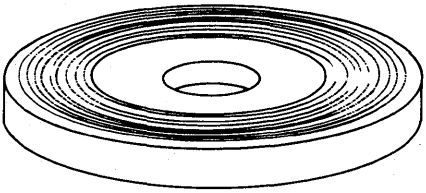
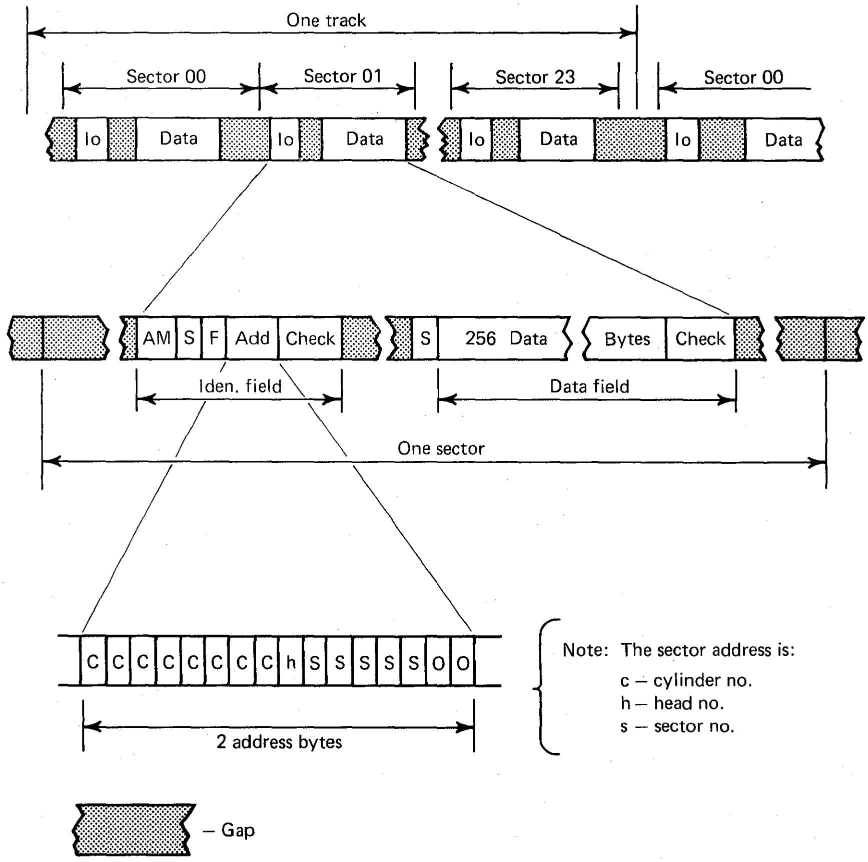


Figure 18.3 Disk storage track arrangement.

Because the read/write heads are all attached to the same carriage, they move across the disk surfaces together. At each access position, two or more tracks are available for reading and writing, those on the top surfaces and those on the bottom surfaces. The tracks thus vertically aligned are considered to form a cylinder. Information can be accessed from any track in the cylinder without moving the read/write heads. An index pulse indicates the start of each track and aligns the start of tracks on a cylinder on any one disk.

Cylinders are numbered 000 through 102 or 202. Cylinder 000 is nearest the periphery of the disk, cylinder 202 nearest the spindle. Cylinders 001, 102, and 003 contain alternate tracks for storage of data should any other tracks be defective. The address of a track is defined by cylinder and head number.

Each track is divided into 24 sectors of equal length (Figure 18.4). Each sector contains a data field and an identifier field.



AM – Address mark (2 bytes)
S – Sync. byte (1 byte)
F – Flag (1 byte)

Add – Sector address (2 bytes)
Check – 3 check bytes
Id field – Identifier field

Figure 18.4 Disk sector storage arrangement.

Sectors, data fields, and identifier fields are separated by gaps. The gaps contain no data; their length varies according to their location within the track. Any given data record can be located by addressing cylinder, head, and sector number.

Data Field. The data field of a sector contains one synchronizing byte, 256 data bytes, and three check bytes. The synchronizing byte has a fixed bit pattern to synchronize the read/write circuits with a file control unit. The 256 data byte positions are reserved for record storage. The record may be arranged in any format to meet the requirements of the data processing procedure. If record lengths are less than 250 bytes, the unused portion of the field should be filled with any valid bit pattern. For operations of reading and writing, records are therefore considered to be of fixed length.

The check bytes verify the information stored in the fields to which they are appended. The bytes are computed by the file control unit when data is transferred to disk storage. Subsequently, when data is read from the disk, the file control unit recomputes the check information and compares it with the check bytes read from the corresponding data field. Any difference in the comparison indicates an error.

Identifier Field. The identifier field contains the physical address of the sector. The start of each sector is denoted by an address mark with two bytes of specially recorded information that is uniquely identifiable. A synchronizer byte is recorded after the address mark. The identifier field contains a flag byte, two address bytes, and three check bytes. The address defines the cylinder, head, and sector numbers.

Flag Byte. Occasionally, because of damage to the disk or a defect in the recording material, a particular track will not be usable. The condition of the flag byte in the identifier field indicates that either (1) the track is a normal data track, (2) the track cannot be used because of a defect, or (3) the track is an alternate for a defective track. Normally, all eight bytes of the flag byte are set to zero when the sector address is first written. Setting certain positions to ones will *flag* the machine that one of the above three conditions exists. Significance of the flag byte coding for the 5444 is shown in Figure 18.5.

When a track is found to be defective, an alternate track is assigned. On the *defective* track, the address of the alternate track is written into the identifier field of every sector. On the *alternate* track, the address of the defective track is written into the identifier field for cross-reference.

Flag byte bit	Setting	Function
0 through 5	0	Not used
6	0	Operative track
	1	Defective track
7	0	Primary track
	1	Alternate track

Figure 18.5 Flag byte significance.

When an access operation is performed to a defective track, the flag byte indicates that the track is defective. The address of an alternate track is given in the identifier field. The read/write heads then move to the new track address where the address of the original or defective track is given in the identifier field. For operating purposes, the read/write heads are now positioned at the correct location as defined in the address instruction.

Alternate tracks are assigned one at a time. Facilities in the system automatically insure that the proper exit procedure and return procedure to the original location are made if a data operation overlaps.

Disk File Organization

Data sets in disk storage may be organized in a number of ways. Three common methods are:

Sequential
Direct
Indexed sequential

Sequential. The records in a sequential data set are consecutive according to a given sequence. Sequential organization is the only one possible for magnetic tape, printer, and punched-card records. Normally, no record in the file can be read or written until all preceding records in the sequence have been read or written, but special facilities are available to permit arbitrary positioning. On tape, records cannot be lengthened, shortened, or deleted without rewriting the subsequent portion of the file.

This same sequential file organization can be used for disk storage. Records may be written sequentially beginning at a given address and continuing through successively higher addresses until the entire file is written. A record control field or *key* may be used to determine the relative location of a record within the file.

Sequential disk storage is useful in those procedures where direct access to particular records is not required; for example, the storage of programs, subprograms, routines, and entire programming systems. Also, sequential storage organization may be used to form a queue of records or data sets waiting to be printed, punched, or transmitted over communication lines as output.

Direct. Records in a file are assigned directly to specific storage locations. In this type of file organization, the addresses in storage must bear a direct relationship to control fields or keys in each record. For example, the assigned part numbers of inventory records might correspond with the storage addressing scheme. Thus, the record for part number 1 is stored in location one, part number 2 in location 2, and so on.

Regardless of the order in which records go into the file, each record always occupies a specific disk location. Therefore, missing or unavailable records produce blank storage locations. A schematic of direct file organization is shown in Figure 18.6.

Indexed Sequential. Neither the sequential nor the direct file organization allows convenient access to any particular record without first reading the preceding records. The indexed sequential file organization overcomes this difficulty by associating the key of each stored record with its corresponding storage address. The keys, with their locations, are then stored separately as a table or *index*. The index contains the key of every record in the file and its address in storage. The data file may or may not be in sequence when it is stored. If it is not, the index can be sorted by key later.

For example, assume records 075, 024, 253 ..., 046 are placed in storage in locations D1, D2, D3, ..., D99 as shown in Figure 18.7. As they are stored, the computer is programmed to compile a table including each record key with its corresponding address. The table appears as follows:

Key	*Disk Address*
075	D1
024	D2
253	D3
.	.
.	.
.	.
046	D99

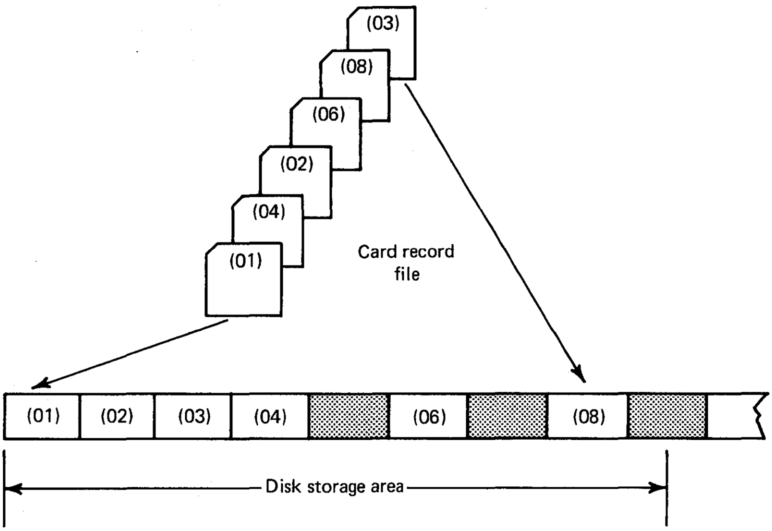


Figure 18.6 Direct file organization.

The table is now sorted by key so that the index appears:

Key	*Disk Address*
024	D2
046	D99
075	D1
.	.
.	.
.	.
253	D3

To retrieve any record from the file, the key must be matched against the index to obtain the corresponding address. This organization has the advantage that storage can be filled to capacity as records are received, and that new records can be inserted without rewriting the entire data set. Also, if records of known length are stored, new records can update or replace old ones as transactions occur, provided the index is modified accordingly.

Applications Processing Programs

Complete programs are also available from IBM and other vendors to process specific applications on the System/3. Programs may be tailored to the needs of a specific industry, a general accounting

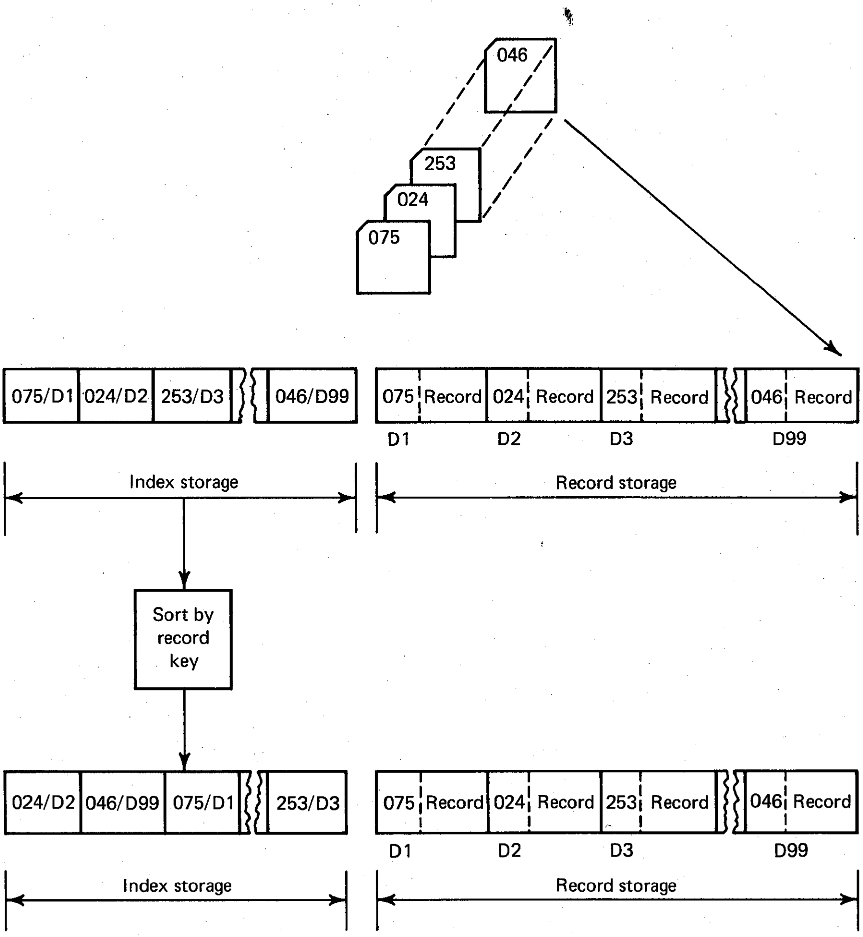


Figure 18.7 Indexed sequential file organization.

function, or to a management system. Users of such programs are generally required to construct their files and other records in some standard manner in accordance with the formats handled by the program.

Typical products offered include such programs as hospital accounts receivable, hospital patient billing, apparel business control, utility billing, and so on. The IBM Apparel Business Control (ABC) system is an excellent example of ready-made programs available to System/3 users. The following features are offered:

Order Edit. This function is performed by two separate programs, order edit and pricing. The order edit program checks new orders for valid product codes, crossfoots the order detail cards, and balances the totals from the order detail cards against the totals in other miscellaneous data cards. The pricing program validates the style and color numbers against those actually carried, and prices the order detail cards.

Order Writing. This program prepares printed orders for use as internal documents and as order acknowledgments to the customer. An order register program controls the machine to print out a register of all orders received.

Bookings Reporting. This program prints a combined bookings and over and under report by style, color, dimension, and size. The report includes a summary of sales received since the last report, a detailed listing of key orders received since the last report, a summary of remaining orders received, a summary of all outstanding or completed cuts, and an over-under summary.

Fabric Requirements Reporting. Two programs produce output that shows seasonal fabric requirements based on either actual or projected sales.

Finished Goods Requirements Reporting. This program produces a report showing finished goods requirements by time period, based on the start shipping and complete shipping dates specified on the orders.

Stock Allocation. This program matches outstanding orders against completed cuts. Cards representing allocated orders details are selected into separate stackers on the MFCU for review and shipment.

Invoicing. This feature is made up of three programs to crossfoot order detail cards, extend and total invoices, print an invoice register, and punch invoice summary cards.

All of these programs can be tailored to the user's specific requirements by filling out a questionnaire. Responses to questions are keypunched into 96-column cards that are used as input to a special modifier program. The modifier program then punches out modified program instructions for the system application programs. That is, the System/3 is used to generate the system programs properly modified to suit the individual user.

A specific configuration of equipment is also required to operate the ABC system.

Questions and Exercises

1. What are the advantages of the utility programs to the computer user?
2. Trace the card paths during a System/3 sort/merge operation.
3. Compute the approximate sorting times for files of 4,200, 6,850, 2,225 and 12,900 cards. (Use Model A2.)

4. Under what circumstances is it justifiable to use the computer system for keypunching and verifying?
5. Under what circumstances can the System/3 convert from 80-column to 96-column cards?
6. What is a disk cartridge and how is it used?
7. Describe the physical arrangement of the disk for data storage.
8. Explain the three main types of disk file organization: sequential, direct, and indexed sequential.
9. How do application processing programs differ from utility programs?

PROGRAM PREPARATION 19

The preceding sections described various components of the System/3 with particular regard for function and system organization. It was shown that the computer operates only under control of a stored program contained in memory. The program completely details a procedure, step by step, in terms of the 28 basic operations the machine can perform. Each step is called an instruction. By executing instructions in the proper sequence, the computer can produce results from available data input. Two arithmetic instructions were described as examples of how the processor manipulates data.

The program is developed for each different procedure by a *programmer.* Programs are read into memory like any other input data and are retained there for the duration of a particular task. When a task is completed, a new program can be stored in preparation for the next procedure to be followed. Transition from job to job can be made automatically if *supervisory programs* are available.

This chapter will deal more completely with the development of the stored program and will lead into the use of a *programming system* called RPG II which is especially designed for the System/3.

General Considerations

It is always surprising to realize how difficult it can be to describe even the simplest task to someone who has little or no preknowledge of what is required. It is especially difficult if the task is to be detailed in writing. As an example, try to write down step by step instructions for tying a shoelace or a necktie.

When giving instructions verbally, a number of assumptions are usually made. For example, it is assumed that even a child knows what a shoelace is, why it must be tied, and where it is used. But, depending upon the person receiving the instructions, none of these assumptions may be valid. Imagine telling—not showing—a barefoot aborigine how to use shoelaces!

Such an illustration may be a ridiculous exaggeration, but the same principles apply to the requirements for describing to a computer exactly how a procedure is to be carried out. Nothing can be assumed. In the case of the computer, however, the capabilities of the machine and the extent of its resources are completely known—such facts are not always available when instructing people.

First, the programmer or systems engineer must take into account the overall aspects of the application which is to be mechanized. This "big picture" of the task can be viewed in three main parts:

Problem definition
Problem analysis
Procedure design

Problem Definition

The application must be understood completely, particularly in terms of *what* results are required and *when* they will be needed. Then it must be determined if proper input is available and whether or not this information can be used to produce the required result on schedule.

Problem Analysis

In most commercial applications, this is primarily the development of an economic justification. There may be many tradeoffs among requirements, schedules, equipment, and implementation costs. For those jobs that *must* be done, like payroll, accounts receivable, and billing, the objective is almost entirely to produce results as economically as possible. For applications like sales analysis, inventory control, and cost analysis, the return on the data processing investment is the prime consideration. For example: What savings are possible if management can know the inventory position by the fifth of the month instead of the fifteenth? Do these savings offset the additional costs of accelerating the schedule? What savings can be made if the inventory position is known weekly instead of monthly?

Procedure Design

The preparation of a diagram or *flowchart* is a basic approach. The flowchart may show the overall aspects first to assist in the problem analysis. Later, the proposed procedure is diagrammed in complete detail.

Figure 19.1 is a system flowchart of an inventory control procedure. The chart uses specific symbols for both processing and input/output functions. The procedure involves a multiple warehouse system. Items are stocked in a central warehouse for distribution to remote warehouses. All customer orders are received in the remote locations and transmitted by teletype (communication link symbol) to the central data processing installation. The system provides four major groups of operations:

a. Updating stock status, based on actual transactions. Shown on the flowchart as Run S1.

b. Response to inquiries from auxiliary warehouses and the central warehouse. Shown as Run O1.

c. Reorder analysis, including purchase order preparation. Shown as Runs P1, P2, and P3.

d. Weekly analysis reports to show slow-moving items, major changes in usage rates, behind-schedule deliveries, economic lot sizes, etc.

All of the above planning and analysis is usually carried out independently of the computer and the programming system.

Program Preparation

To develop a final program, the programmer must be concerned with at least some of the following in direct relation to the machine and the programming system with which he is to work.

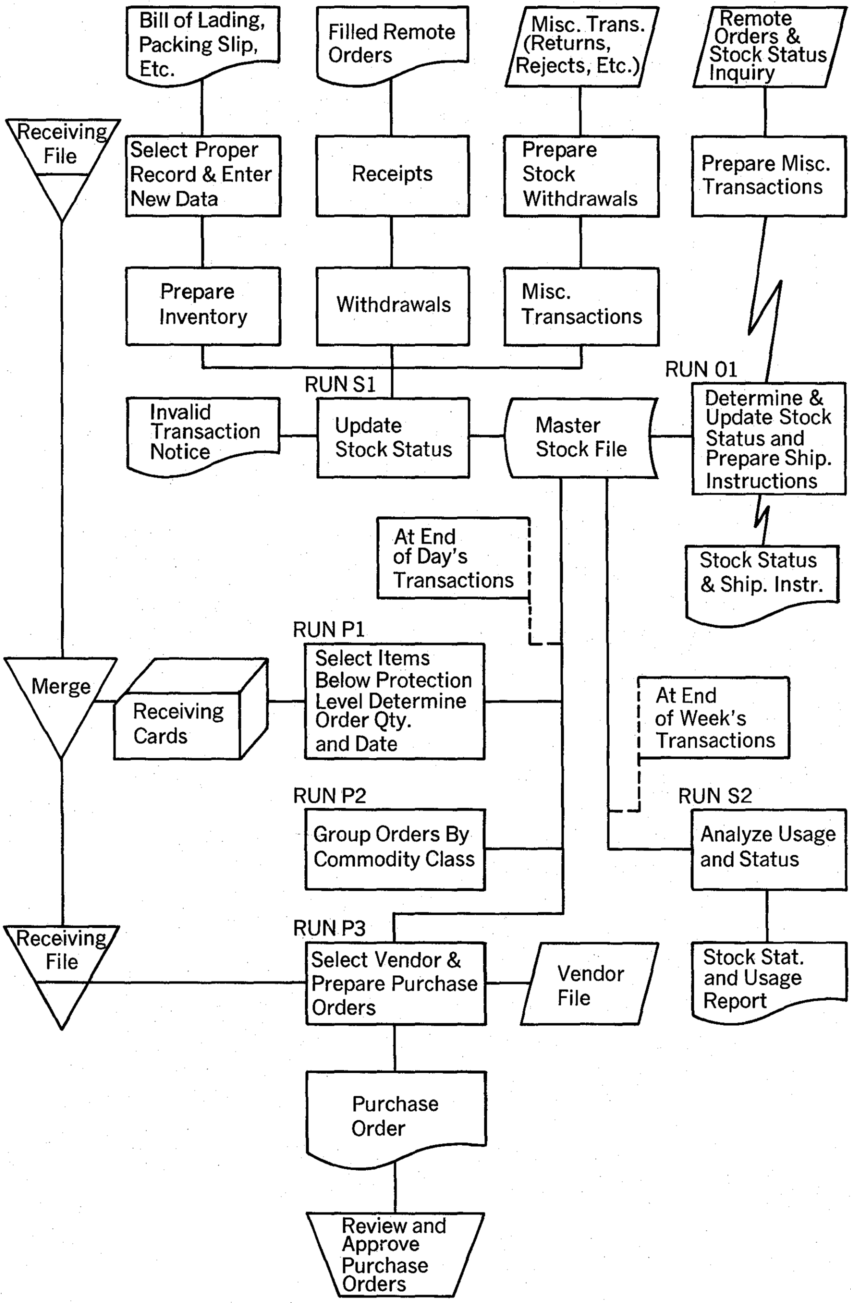

Bill of Lading, Packing Slip, Etc.
Filled Remote Orders
Misc. Trans. (Returns, Rejects, Etc.)
Remote Orders & Stock Status Inquiry
Receiving File
Select Proper Record & Enter New Data
Receipts
Prepare Stock Withdrawals
Prepare Misc. Transactions
Prepare Inventory
Withdrawals
Misc. Transactions
RUN S1
RUN 01
Invalid Transaction Notice
Update Stock Status
Master Stock File
Determine & Update Stock Status and Prepare Ship. Instructions
At End of Day's Transactions
Stock Status & Ship. Instr.
RUN P1
Merge
Receiving Cards
Select Items Below Protection Level Determine Order Qty. and Date
At End of Week's Transactions
RUN P2
RUN S2
Group Orders By Commodity Class
Analyze Usage and Status
RUN P3
Receiving File
Select Vendor & Prepare Purchase Orders
Vendor File
Stock Stat. and Usage Report
Purchase Order
Review and Approve Purchase Orders

Figure 19.1 Inventory control procedure.

1. Conversion of original data to the input medium. With the System/3, the preparation of 96-column cards is involved. A small amount of data can also be entered with the printer-keyboard.

2. Allocation of the computer storage areas to data, instructions, and related information. Storage capacity directly affects the method of job execution; that is, the number of records stored at any one time, the length of the program, and consequently the extent of the processing that can be completed in any one pass of the data through the machine.

3. The availability of reference data: tables and constant factors. Such information may be required throughout a job run but cannot be conveniently read in with each input record. Such information must be given space in memory.

4. Requirements for accuracy. These vary with each application. For example, a payroll must be 100% accurate. On the other hand, some tolerance for error may be permissible in a sales analysis where trends rather than actual figures are to be shown.

5. Ability to restart the system when an unscheduled interruption occurs, such as a power failure. Usually, work is performed in manageable batches with controls and totals established by each batch. If an unexpected work stoppage occurs only the last batch must be redone, not the entire job.

6. Format of output data in the form of punched cards and printed reports.

7. Availability of program routines that have been used and tested in other procedures. Some may be used to advantage in the current procedure being developed. Many programming systems make provision for a *library* of subroutines or programs that are available. Means are provided to properly insert a library item in a new program automatically. The programmer *calls out* an item by writing down proper identification in his program.

8. Resource management. Consideration should be given to recording the utilization of the system, that is, logging the running time for each job, recording exceptions to procedures that cannot be processed, planning for system maintenance, and allowing for unscheduled downtime and diagnosis.

9. Housekeeping routines. Although these routines are usually an inherent part of the programming system, there must be pro-

vision when starting a job to be sure all *housekeeping* has been done first. That is, indicators may be preset, storage areas cleared, and constant data moved or arranged according to instructions. In case of restart, additional housekeeping may be required to restore the system to a ready state to reprocess the last batch of work not completed.

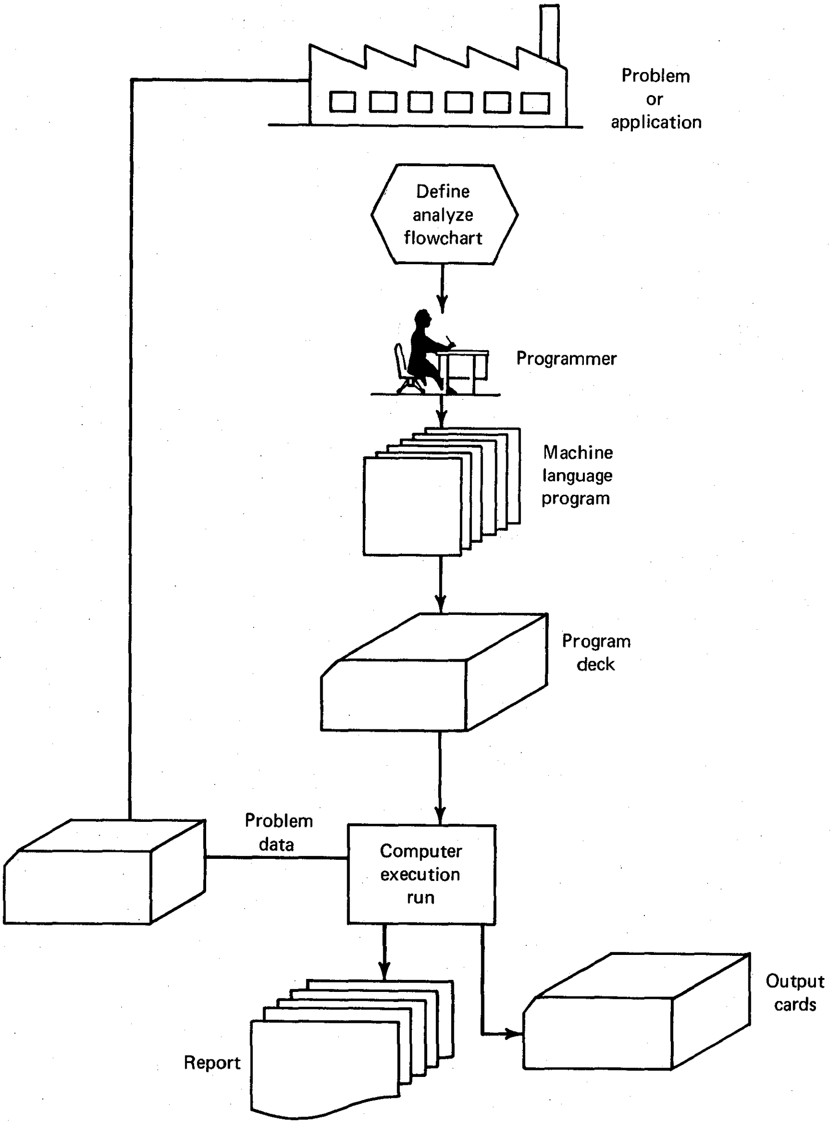


Figure 19.2 Relationship between the computer and programmer. Actual machine language coding.

There may also be instructions to the operator printed out on the printer-keyboard telling which form to insert in the printer or what cards to put in the MFCU. Additional messages may be printed as entries in the log of system utilization.

Figure 19.2 shows the relationship between computer and programmer if it were practical to write a program in actual machine language. The problem is first analyzed, defined, and diagrammed as described earlier. Then the programmer writes the program and supplies all tables, formulas, codes, or other reference materials needed for the specific application.

When the program is prepared, the problem then becomes input data and the computer produces the planned output. A number of difficulties make this approach entirely impractical:

1. All instructions must be coded in machine language. Even though the System/3 is a relatively simple configuration of units, the production of instructions of varied format in hexadecimal code would be very difficult indeed.

2. All instructions must be arranged in the exact sequence in which they are to be executed by the computer. If one or more are omitted by error, all succeeding instructions must be relocated in storage to make room for insertions. Otherwise, some branching technique must be used to include new instructions. This clerical accounting for all storage space must be carried on entirely by the programmer.

3. The full burden of logic and program organization is placed directly on the programmer.

4. Previous experience (i.e., tested programs that might be used for part of the procedure) is difficult to work into the new program. Such programs must be linked to the new program by additional handwritten instructions.

5. The programmer must understand the computer in detail. He must know each indicator and register and he must program their functions entirely.

6. To test the program for error, all checking must be done in machine code. If corrections are required, the clerical work of modifying and rechecking becomes exceedingly complicated and time-consuming.

Program Assembly

Early in the history of computer development it became obvious that methods must be worked out to free the programmer from the burden of machine language coding. As engineering sophistication of machines or *hardware* advanced, there was a corresponding need for the techniques of programming or *software* to keep pace. Otherwise, full utilization of future machines could not be realized.

Because a large proportion of the task of program writing proved to be clerical, it seemed logical to attempt the assignment of such work to the computer itself. The machine might thus do a large share of the work in preparing its own program. Consequently, in the evolution of computer application, a number of software systems have been developed. The simplest concept centers around the principle of symbolic programming, or *assembly.*

In this approach all data, records, fields, and related information to be used in the procedure are labeled with symbolic identifiers when the program is originally written. No specific storage locations are assigned. That is, each item of data to be addressed by an instruction operand is given a name to easily identify that item for the programmer.

The *amount* of storage required is specified by indicating the length of each item on the written program sheet. This will later tell the machine exactly how much memory space will be required, but not where the space will be located for the actual program. For example, an input pay record might be described as follows:

Label	*Length*
PAY PER	2
DEPT	3
LOC	2
SHIFT	1
MAN NO	5
RATE	3
REG HRS	3
OT HRS	3
.	.
.	.
.	.
BLANK	10

The sum of field lengths within a record must equal the total length of the record. Forms for writing the program in longhand are provided to standardize the format and to enforce the established conventions of the system. For example, labels may be restricted to 10 spaces; length of field to three spaces. Means are also provided to address an entire record as well as individual fields and items.

Next, a standard abbreviation is given each operation the machine can perform. The abbreviations take the form of *mnemonics*: ADD, SUB, DIV, ZAZ (Zero and Add Zoned), SZ (Subtract Zoned Decimal), etc. The use of mnemonics is intended to make the programming language as much like everyday English as possible. As a result, the language is designed to be easy to learn and relatively simple to use. The programmer writes these standard mnemonics in instructions just as if they were actual machine coded binary notations.

Instruction operands address the previously named record fields and reference items. Thus, a small program routine to calculate gross pay might appear as follows:

```
Label    Operation   Operand     Comments

START    SEL         READER      Select input device
         RD          PAY REC     Read one record into storage
         CLA         REG HRS     Put regular hours in accumulator
         ADD         OT HRS      Regular hours + overtime hours
                                 = total hours
         MPY         RATE        Total hours X rate = gross
         STO         GROSS       Put gross pay in record
          .            .
          .            .
          .            .
         SEL         PRINTER     Select output device
         WR          PAY REC     Write one record
         BR          START       Branch to repeat routine
```

This example is simplified for purposes of illustration and is not an actual sample of any particular language. Several basic concepts are shown.

First, the programming sheet is laid out to include at least four items of information: label, instruction operation part, instruction operand, and comments. The comments have no effect on the procedure but are used as required for clarification purposes only. The comments also make it easier for others to follow the procedure in case changes or alterations must be made later.

The first step of the program is labeled START as a reference point that may be addressed by other instructions. The last instruction operand addresses the first instruction label indicating a branch back to the beginning of the routine. Thus, the portion of the program shown forms a complete *loop* that is repeated for each record of the input file. Only an instruction to be addressed is given a label.

After the program is completely written and all items of data properly identified, a single entry is made indicating the actual storage location of the first instruction. The completed program with all related data is now punched, line by line, into cards. The resulting deck is held for input to a computer assembly run as a *source program.*

The computer is loaded with the assembly program which has been previously prepared as part of the system. In this run, the source deck is processed as data. All symbolic information is converted to actual machine coding. Essentially, two passes of the source data are required, although the system is often designed to produce a complete *object program* in one run.

1. From the beginning memory location in the source program, the machine can determine the actual locations of all succeeding instructions by progressive addition. The length of all record fields and constant factors is also given in the source program. Again, by addition of the lengths, the computer can assign an actual memory location to each item.

All symbolic labels with corresponding actual locations are stored as a table in memory.

2. After all actual locations have been determined, instruction operands can be converted by an operation of table look-up. The assembly program also stores as a reference table all possible mnemonic operations with their machine code equivalent. During the processing, these symbols in the source program instructions are also translated into machine codes.

The converted or assembled object program can be punched out in cards as an *object deck.* A printed list of the source program including the machine coded translation can be obtained as a by-product of the assembly.

Programming Systems

Figure 19.3 shows the relationship between computer and programmer when a programming system is used. The system includes at least two parts. First, there is the system language with a stan-

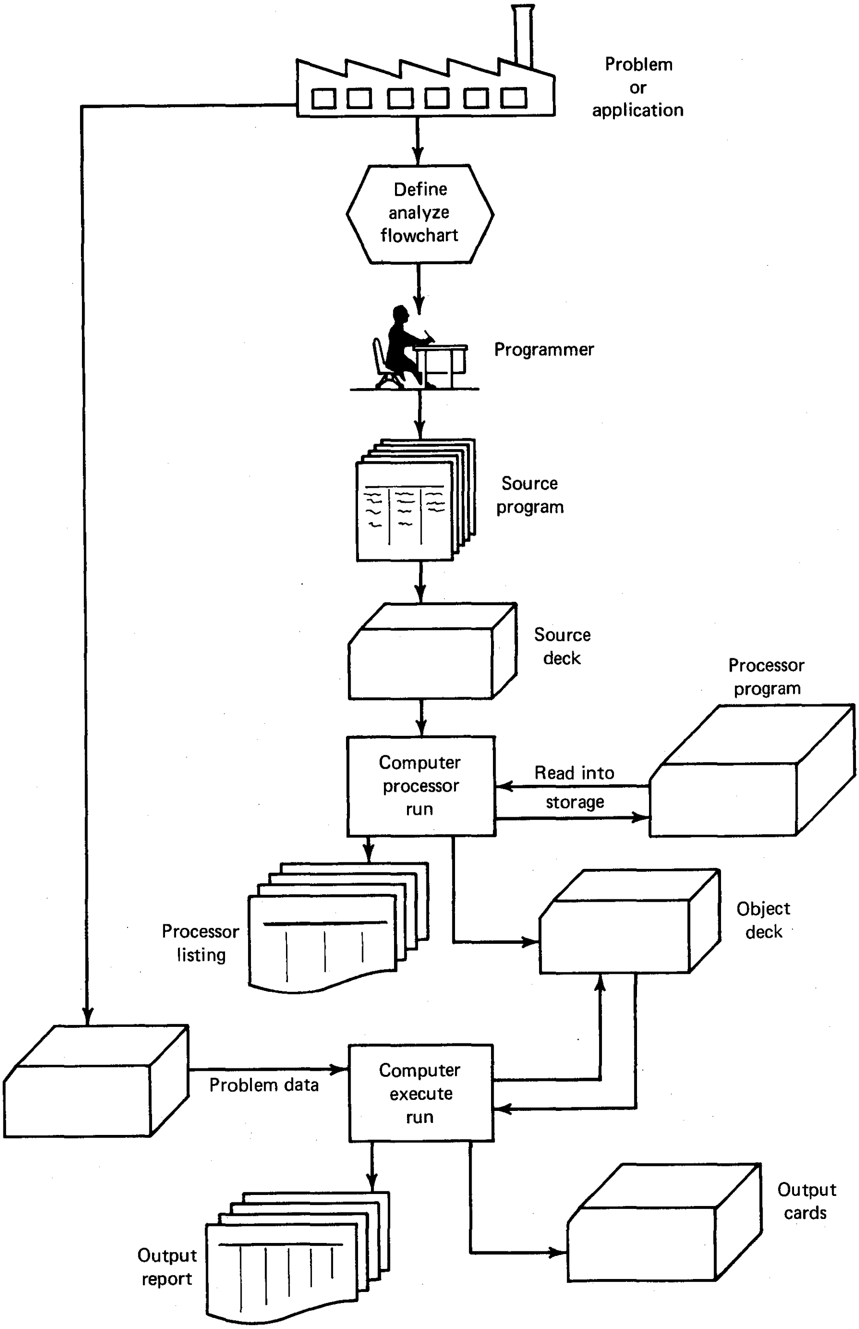


Figure 19.3 Relationship between computer and programmer. Programming system.

dardized convention of grammar and syntax. This is the language in which the programmer writes the program. Second, there is an assembly program the computer uses to translate from the system language to another machine language program.

The input to the assembly, as written by the programmer, is called the source program. It states the requirements of a given task and the method of solution. As previously stated, before the programmer writes his source program, he must have completely analyzed and defined the problem.

The output from the assembly is the object program. This is the translation of the source program to the machine language of the computer system on which the object program will be used. The assembly program is available in cards and thus can be loaded into the computer for the assembly run.

The source program is keypunched directly into cards directly from the programmer's original written sheets. These cards become input data to the assembly. When all the source information has been translated, the resulting object program is available as any normal output from the computer. Usually, the object program is listed to record the assembly process. Exceptions or errors preventing assembly are printed as the source program is being translated. Unless severe trouble is encountered, the program is punched into output cards as an object deck.

During the assembly process a number of checks can be made on the programmer's work. For example, the programmer may have referred to nonexistent parts of the program or may have forgotten to make reference information available. Any illegal mnemonics for operations can be listed, grammatical violations can be noted, and out-of-sequence instructions can be shown. Sometimes a controlled tolerance for error is built into the assembly program in such a way that source information is used even though misspelled or otherwise in error.

Once the object program is properly assembled and punched, it can be loaded into the computer storage for an execution run. Normally, this first run is to test the object program with sample input data. The test data should be arranged to cause the machine to execute all possible operations and to follow each exception to the procedure completely. If the program is fairly complex, it will be rather unusual for this test run to be executed perfectly.

The process of removing errors, or *debugging* the program, is carried out next. Many programming systems provide special aids to locating and fixing errors with printouts, tracing routines, and other diagnostic methods. One method of error detection halts machine operation at the point of error and prints out the entire contents of memory. The programmer may then examine this "snapshot" of storage for some clue to the malfunction. Meanwhile, the machine can be performing other work.

When all errors are finally located, correction cards are punched and inserted in sequence in the source deck. The deck is reassembled and the test is rerun on the new object program. This program is debugged if necessary, reassembled and tested again until a final, error-free object program is obtained. The proven and tested object program can be used again and again with varying sets of task data.

Compilers

The preceding discussion has described one method of overcoming the language barrier between computer and programmer. With this method, programs can be written in terms more easily understood (and learned) by the programmer. At the same time, the assembly does much of the clerical work of program preparation. The assembly process also performs some auditing functions, thereby reducing the occurrence of many common manual errors and making easier the job of testing and correcting the operation of the object program.

When an assembly system is used, the programmer writes all the detailed computer steps. And even though there are many advantages to the method, he must furnish all instructions on a one-for-one basis. For one instruction written in the symbolic language, the assembly will produce only one instruction in machine language coding.

The next step in programming system development is to increase the effectiveness of the assembly by enlarging its functions. For example, many phases of procedures are duplicated or repetitious. All programs require routines to read and write records, to check for errors, and to perform many standard calculations of arithmetic. Many procedures also use standard methods of converting data from one form to another.

Common routines can be included in a table, array, file, or *library* that is available to the assembler. The programmer is then given the option to instruct the assembly program to select appropriate routines from the library and insert them in the object program as specified. The operation part of library instructions is standard, but the addresses must be modified to reference the location of data as defined by the source program.

To make these enlarged functions of the assembler available, additional mnemonic operations can be added to the language. For example, assume that a tested routine for extracting square root is stored in the library. This routine may be given the name SQRT. If the programmer needs the square root routine as a part of his procedure, the mnemonic SQRT is written on the source program sheet as the operation part of a symbolic instruction. The operand designates the storage location of the number and the location where the root is to be stored.

When the assembly program encounters SQRT, reference is made to the library rather than to the table of machine operation codes. The SQRT subroutine is then assembled into the object program with addresses obtained by the regular assembly process. In this case, a number of machine instructions are produced in the object program for one instruction in the source program.

To make more and more functions available, the size of the library can be increased. As the resulting programming language becomes more flexible, it also becomes more independent of actual machine operations. Consequently, there is the effect that the programmer now thinks in terms of the programming system rather than in terms of actual computer functions related to a particular machine.

In fact, as the language becomes more and more sophisticated, the programmer is required to know less and less about the computer system. Or, as the language is developed to a higher and higher *level*, it tends to become *problem* oriented rather than machine oriented. Thus, the language is a more effective aid in expressing the problem while the computer does a greater portion of the conversion from procedure to machine operations.

Functions can be added to the system to link entire program routines together and to automatically include housekeeping, checkpoints, restart, and control routines. The program assembler now becomes a *compiler* with the ability to search the library and compile an object program from given specifications and available routines.

Finally, the language can become standardized among a large number of users so that there is general agreement as to the rules of grammar, syntax, format, limitations, and so on. At the highest level, the language is completely independent of any particular computer system. Compilers can be developed to translate an original source program for use by any computer, even those machines which are produced by a variety of manufacturers.

Therefore, programs written in the standardized language can be compiled on a number of different computer systems. Of course, the necessary configuration of equipment must be available with sufficient storage capacity and proper input/output devices.

Several such high-level languages are widely used throughout the computer industry.

COBOL

This COmmon Business Oriented Language is based on a system first developed in 1959 by a committee of government users and computer manufacturers. The language itself, as well as the techniques for using it, are concerned mostly with commercial problems and results.

The COBOL language bears little resemblance to machine language. And the problem programmer has little to do with the method by which his COBOL program is translated into machine language.

The following example illustrates the basic principles of the problem oriented type of programming system. Assume that the value of an item called INCOME is to be increased by the value of an item called DIVIDENDS. COBOL can specify this addition by the following sentence in the source program:

```
ADD DIVIDENDS TO INCOME.
```

Before the COBOL processor can interpret this sentence, however, it must be given certain additional information. For example, the programmer must write DIVIDENDS and INCOME in a special part of the program called the *data division.* The facts about the data represented by those names are stated here, such as maximum size, how the data is expressed, etc.

When the compiler encounters the sentence, it also has access to certain information that will aid it in translating the sentence. In addition, it can obtain certain information built into the processor itself. The exact procedure varies from machine to machine. In any case, the problem programmer is not directly concerned with the details.

First, the compiler examines the word ADD. It then consults a special list of words that have clearly defined meanings in COBOL. This list is part of the compiler. If ADD is one of those words, the compiler interprets it to mean that it must insert into the object program the machine instruction or instructions necessary to perform addition.

The compiler then examines the word DIVIDENDS. Since it can obtain certain information about DIVIDENDS, it will know where and how this information is to be stored in the computer, and it will insert into the object program the instructions needed to locate and obtain the data.

When the processor encounters the word TO it again consults the special word list. In this case, it finds TO directs it to the value of INCOME which is to be increased as a result of the addition.

The compiler must now examine the word INCOME. Again it has access to certain information about this word; as a result, it can place in the object program the necessary instructions for locating and using INCOME data.

Notice that the programmer placed a period (.) after the word INCOME, just as he would in terminating an English-language sentence. The effect of the period in the COBOL compiler is quite similar. It tells the compiler that it has reached the last word to which the verb ADD applies.

The steps just described are performed by the compiler in creating an object program. They may not be performed in exactly this way or in the same sequence, because machines vary and because each compiler is adapted to a particular machine. Regardless of the machine, the same COBOL sentence produces machine instructions that will cause the object program to add together the values DIVIDEND and INCOME.

FORTRAN

FORTRAN is an acronym for FORmula TRANslation system. The language is made up of individual commands or *statements* of a program, operators (such as + or —), and expressions (such as $C = A + B$). The compiler is a program for the computer which tells it how to translate the program written in FORTRAN into a program in machine language.

The FORTRAN language is intended to be capable of expressing any problem of numerical computation. In particular, it deals easily with problems containing large sets of formulas and many variables, and it permits any variable to have up to three independent subscripts.

Each FORTRAN source program is composed of a number of statements. Each statement deals with one aspect of the problem or procedure. For example, a statement may cause data to be fed into the computer, calculations to be performed, decisions to be made, results to be printed, and so on. Also, some statements written by the programmer do not cause any specific computer action, but rather provide information to the processor. For example, the following two statements are listed with their effects:

READ 1, A This causes the computer to read a card and handle the data in such a way that if the card has a value or quantity 97.0 punched in it, then A will have the value of 97.0.

C = 3.*A The asterisk indicates multiplication. Thus, this statement means multiply A by 3.0 and set C equal to the result. Using the data of the previous example, C would be given the value 291.0, the product of 3.0 X 97.0.

Notice that when the source program is executed, the computer is not instructed merely to find the value of C. It is given the data (in this case 97.0), instructed how to make the computation, and told what to do with the result.

An important aspect of FORTRAN, as of many other programming systems, is that the language can be largely independent of the computer on which the object program is to be executed. But the FORTRAN compiler is dependent upon the object machine because it must produce an object program in that machine's language. For this reason, each object machine must have its own processor. This is another way of saying that an original FORTRAN source program can be translated for use on several types of computers. But the source program must go through the translating process for each type of machine on which it is to be used.

Questions and Exercises

1. What are the three main aspects of a procedure or application that must be considered for mechanization? Explain each.
2. In what detail would a system engineer need to understand machine functions to mechanize the procedure in Figure 19.1? Explain.
3. How does the data capacity of storage affect the method of doing a job?

4. What are tables and reference data and where would these be obtained in a procedure?
5. What is resource management?
6. What is housekeeping?
7. Give some important reasons why programming in machine language is impractical.
8. What is the purpose of a program sheet?
9. What are some important advantages of a programming system?
10. What is the difference between a problem-oriented and a machine-oriented programming system?
11. What are some of the main characteristics of COBOL and FORTRAN?

REPORT PROGRAM GENERATOR, VERSION II 20

The Report Program Generator, Version II, compiles or *generates* a machine language object program for the System/3 from written specifications of a problem or application. Input data files, the planned output, and all operations of data handling are expressed in predetermined format by the programmer. Originally developed for the IBM System/360 as RPG I, Version II has been expanded with additional features for the System/3 that can be used in a wide range of commercial data processing jobs. The RPG system has both a language and a processor.

The language is primarily a set convention that has been established for filling out the various programming forms. Different specification sheets are used to code each aspect of the source program. For example, one form is used only for input file specifications. Another is used only to describe calculation and data handling operations. Headings on each sheet describe the particular use of the forms.

After specifications have been written, cards are punched from the data. Each line on a form is a card in the source deck. This deck and the RPG processor deck are read into the computer by the multifunction card unit during an assembly run. One MFCU hopper contains the source program, tables (if any), and blank cards for the object deck. The other hopper contains the processor, approximately 3,000 cards containing about 40,000 instructions. The assembly is done in six phases requiring about 2,000 bytes of storage per phase. The object program can be produced with two options: (1) as a printed listing only or (2) both in cards and as a listing with all machine instructions to run a planned task. After the assembly run, the object program can be loaded into the computer for a test or execution run with actual problem data.

During assembly, the processor checks many clerical aspects of the source program and produces printed messages to the programmer to identify any errors encountered during production of the object program.

Debug statements may be placed in the source program deck for listing and to display fields and indicators, information that can be helpful for locating programming errors. The contents of any field and any program indicator can be printed. The object program can be generated with or without the debug option.

File Description

In RPG language, the term *file* refers to any logical collection of related records. The records may be on either cards or paper. Therefore, there is the rather unusual connotation that a printed report is also a "file." Three kinds of files are considered in the system:

1. Input files from which information is read.

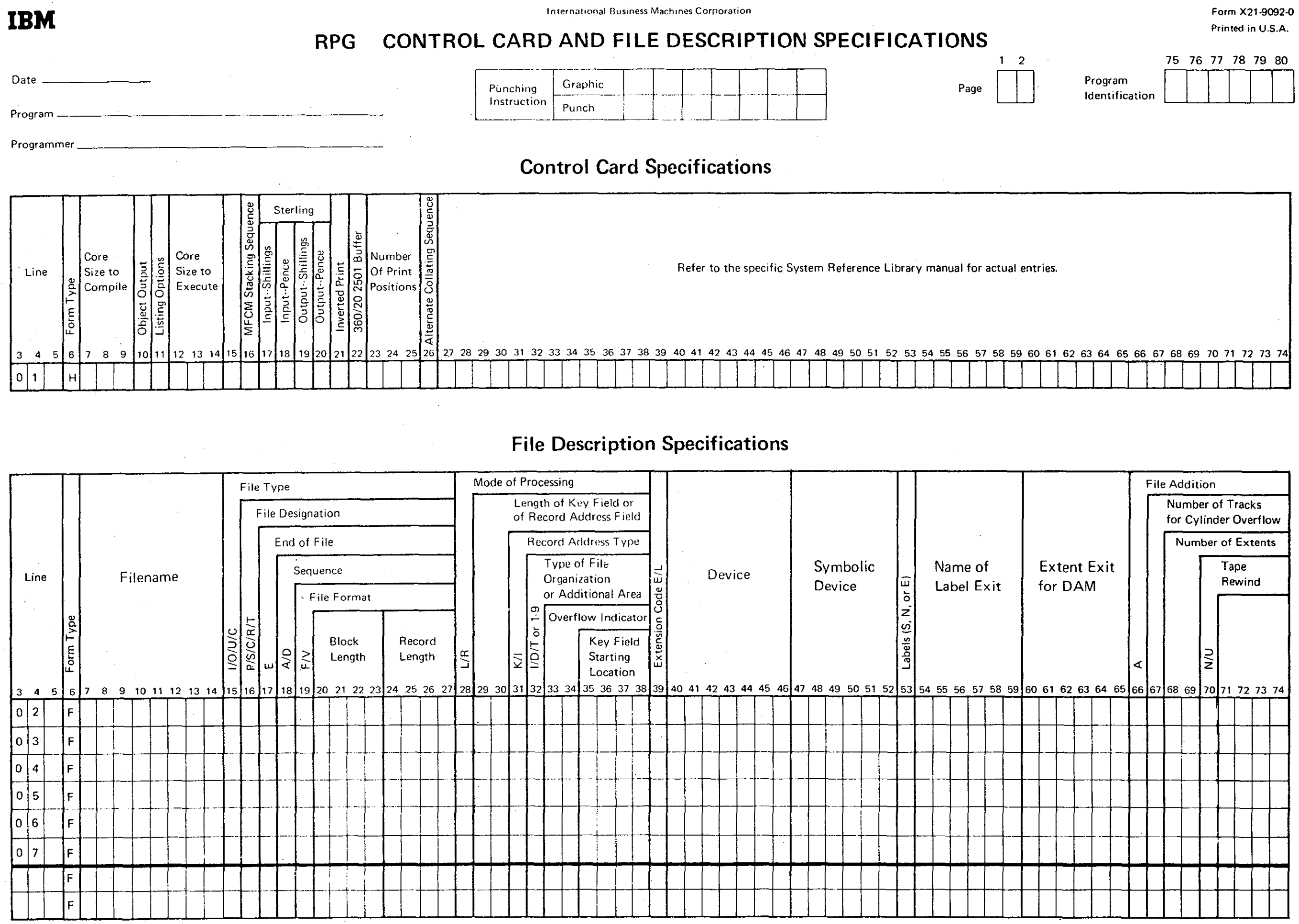
2. Output files on which information is printed, punched, or both printed and punched.

3. Combined files; that is, card decks in which some or all of the cards are to be used for both input and output. Example: two fields in the input deck are to be added and the sum punched in a third field in the same cards.

The RPG programming system is designed mainly to process files of cards.

Specification Sheets

Certain information is common to all of the specification sheets used in the RPG system. Refer to the Control Card and File Specification form shown in Figure 20.1.



IBM

International Business Machines Corporation

Form X21-9092-0
Printed in U.S.A.

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

Date ____________

Program ____________

Programmer ____________

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Control Card Specifications

Line	Form Type	Core Size to Compile	Object Output	Listing Options	Core Size to Execute		MFCM Stacking Sequence	Sterling				Inverted Print	360/20 2501 Buffer	Number Of Print Positions	Alternate Collating Sequence	Refer to the specific System Reference Library manual for actual entries.
								Input-Shillings	Input-Pence	Output-Shillings	Output-Pence					
3 4 5	6	7 8 9	10	11	12 13 14	15	16	17	18	19	20	21	22	23 24 25	26	27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
0 1	H															

File Description Specifications

Line	Form Type	Filename	File Type	File Designation	End of File	Sequence	File Format	Block Length	Record Length	Mode of Processing	Length of Key Field or of Record Address Field	Record Address Type	Type of File Organization or Additional Area	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	File Addition	Number of Tracks for Cylinder Overflow	Number of Extents	Tape Rewind	
			I/O/U/C	P/S/C/R/T	E	A/D	F/V			L/R		K/I	I/D/T or 1-9									A			N/U	
3 4 5	6	7 8 9 10 11 12 13 14	15	16	17	18	19	20 21 22 23	24 25 26 27	28	29 30	31	32	33 34	35 36 37 38	39	40 41 42 43 44 45 46	47 48 49 50 51 52	53	54 55 56 57 58 59	60 61 62 63 64 65	66	67	68 69	70	71 72 73 74
0 2	F																									
0 3	F																									
0 4	F																									
0 5	F																									
0 6	F																									
0 7	F																									
	F																									
	F																									

Figure 20.1 Control card and file description.

Each sheet is divided into 80 vertical columns for convenient punching into cards, either 80- or 96-column. Each horizontal line is divided into various fields for coded entries which will describe the application. The programmer writes one character, digit, or symbol in a column.

Columns 1 and 2 are located at the top right corner of the form and are used to number the pages. Columns 75 through 80 are just to the right of the page identification and serve to contain the program identification. *Columns 3 through 5* are used to number the lines of coding. The first two digits are preprinted on the sheets; the third may be used as an insert digit when items are omitted from the normal sequence. When punched in cards, the line numbers keep the records in order within each page.

Column 6 indicates the type of form so that this information will also be punched in the source deck. The following codes are used:

H	Control or "header" card (because it is the first card in the source deck)
F	File description (cards or printed output)
I	Input specifications
C	Calculation specifications
O	Output specifications
E	File extension specifications
L	Line counter specifications

All type codes are preprinted on the corresponding form. Only H and F are used on the form shown in Figure 20.1. All of the above items are aids to the clerical organization of the coding and have no effect upon the object program other than to insure proper sequencing and identification of the source data.

Figure 20.2 shows the proper card arrangement in the source deck after punching. Cards outlined in dotted lines are optional. For example, a program may be assembled for a card-to-print operation where no calculation is required.

Column 7 may contain an asterisk (*) to identify written comments. The asterisk indicates that all of the following information on that line is not to be assembled as part of the object program. The programmer may use the comment line to document notes explaining the procedure. The notes may be particularly helpful later if the source program is to be revised or changed and must therefore be understood by someone other than the original programmer.

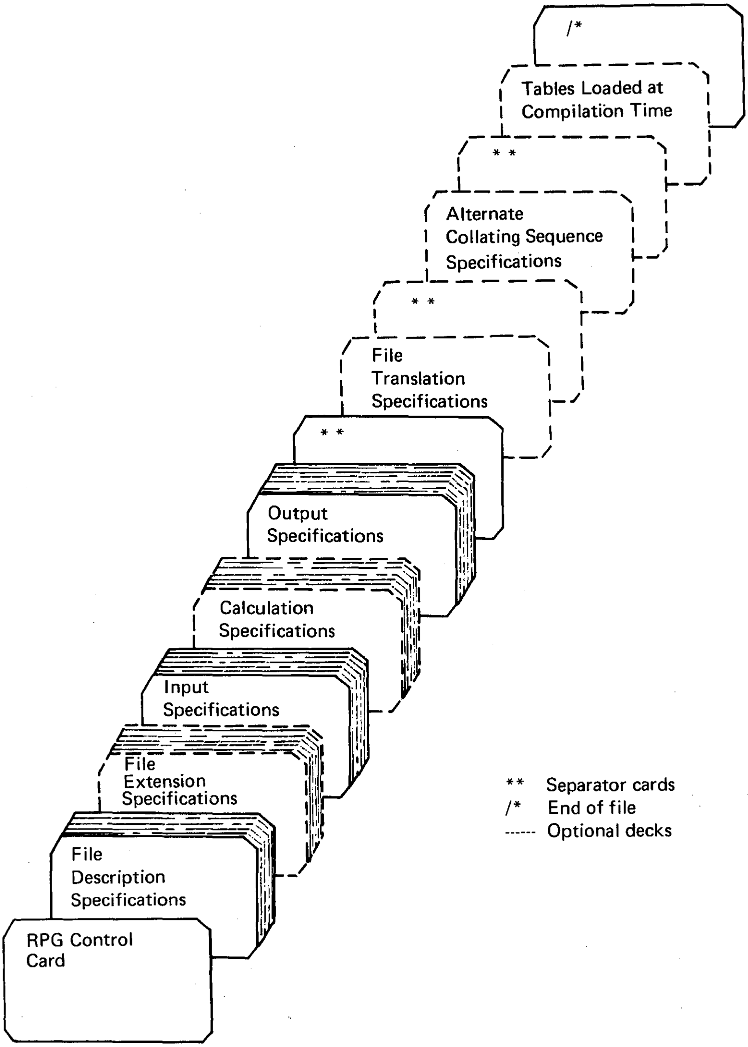

/*
Tables Loaded at Compilation Time
* *
Alternate Collating Sequence Specifications
* *
File Translation Specifications
* *
Output Specifications
Calculation Specifications
Input Specifications
File Extension Specifications
File Description Specifications
RPG Control Card
** Separator cards
/* End of file
------ Optional decks

Figure 20.2 Arrangement of RPG II source deck.

Control or Header Card Specifications

One control card is required for each source deck. The card specifies to the processor a number of general items of information which apply to both the assembly process and the entire application.

Columns 7 through 9 indicate the maximum number of bytes that can be stored by the system used to *compile* the object program as follows:

Entry	*Explanation*
008	8,192 byte capacity
012	12,288 " "
016	16,384 " "
024	24,576 " "
032	32,768 " "

Column 10 indicates whether or not an object deck is to be punched during assembly. An object program can be generated with only a listing. It is sometimes desirable to check for programming errors so that corrections can be made before the final object deck is punched. The following entries are used:

Entry	*Explanation*
Blank	Object program is punched into cards and a listing with diagnostics is given.
E	Object program is not punched into cards, but the listing with diagnostics is given.

Column 11 controls system halts and the printing of the program listing during assembly:

Entry	*Explanation*
Blank	Compilation stops if any programming errors are detected. A program listing with diagnostics is given.
B	Compilation stops if serious programming errors are detected. No program listing is given.

Columns 12 through 14 indicate the maximum number of bytes that can be stored in the system which will *execute* the object program.

Entry	*Explanation*
Blank	The size of the system that will execute the object program is the same size as the system that compiled the source program.

(Other entries are the same as for columns 7 through 9.)

Column 15 is used to indicate whether or not the DEBUG operation is to be performed during assembly.

Entry	*Explanation*
Blank	DEBUG operation is not performed.
1	DEBUG operation is performed.

Column 16 is not used.

Columns 17 through 20 are used to describe the format of sterling fields used in sterling currency. Otherwise, the fields are left blank. The formats will not be explained here.

Column 21 describes the type of punctuation used in numeric constants; that is, numeric information retained for reference by the object program at execution time. The entry also controls format of an UDATE field and the punctuation provided by certain edit codes. This entry will not be explained here.

Columns 22 through 25 are not used.

Column 26 provides for altering the collating sequence of characters. Normally every alphabetic, numerical, and special character holds an assigned position in relation to all other characters. This special order is known as the *collating sequence* and is the basis for all sorting (Figure 20.3). The System/3 uses a collating sequence based on the way in which characters are represented in the machine.

With this entry in the control card specification, the programmer may elect to change the collating sequence to one other than that used by the machine. Or, two or more characters may be assigned the same position in a sequence. This sequence will then apply to the compare operation using alphameric fields and also to match fields. The alternate collating sequence will have no effect upon numerical compares, control fields, or table look-up operations. A Translation Table and Alternate Collating Sequence Coding sheet is used with this entry. Every character that is to be out of the normal sequence must be specified. Cards are punched from the sheet and are placed in the source deck as shown in Figure 20.2.

Columns 27 through 74 are not used.

1	Blank
2	¢
3	.
4	<
5	(
6	+
7	|
8	&
9	!
10	$
11	*
12	)
13	;
14	¬
15	- (minus)
16	/
17	,
18	%
19	— (underscore)
20	>
21	?
22	:
23	#
24	@
25	'
26	=
27	"
28	A
29	B
30	C
31	D
32	E
33	F
34	G
35	H
36	I
37	}
38	J
39	K
40	L
41	M
42	N
43	O
44	P
45	Q
46	R
47	S
48	T
49	U
50	V
51	W
52	X
53	Y
54	Z
55	0
56	1
57	2
58	3
59	4
60	5
61	6
62	7
63	8
64	9

Figure 20.3 System/3 Model 10 collating sequence.

File Description Specifications

Each file to be used in an application must be described in the source program. As previously stated, a printed report is also considered as an output file in RPG language. The programmer enters the proper items on a File Description Specification sheet shown in Figure 20.1. The description serves to:

1. Assign each file of the job a positive identification. Whenever possible, this should be either the true name of the file or an easily recognizable symbol (such as PAYREC, ACCTSREC, INVREC). The assigned name will be used later on other specification sheets to identify the files for data handling operations.

2. Assign an input or output device to every file.

3. Indicate whether the file will provide input data, serve for output data, or both.

4. Specify if an input or combined file is to be sequence checked.

5. Show which files must be completely processed before the job is finished.

6. Show that a file may or may not be used for a specific job.

7. Specify whether additional I/O areas are required.

The specification sheet is designed so that only one line is required to describe one file.

Columns 1 through 6 and *columns 75 through 80* were described in the previous section.

Columns 7 through 14 are used for the file name made up of from one to eight characters. The name can be any combination of digits and letters, but the first character must be a letter. The name must begin in column 7 and cannot contain any special characters or blanks. Every file must be given a name, but no two files can be given the same name. The file must be referred to by its assigned name throughout the program.

Column 15 indicates what kind of file is named as follows:

Entry	*Explanation*
I	Input file
O	Output file
C	Combined file

All files must be further described on input or output specification sheets. If either punching or printing is to be done on the input deck, it must be specified as a combined file. If only one input or combined file is used, it must be placed in the primary hopper of the MFCU.

Column 16 is used to further describe an input or combined file:

Entry	*Explanation*
Blank	Must be left blank for all output files.
P	Primary file, the main file in the job, is used to control the order in which cards from all files are processed.
S	Secondary file, any file not considered to be the primary file.
T	Table file, a file containing only related data items of the same type, length, and decimal positions.
D	Demand file, an input file from which records can be read by a special FORCE operation only. (The FORCE operation can be used to bypass normal RPG record selection.)

A primary file is required for every job but there can be several secondary files. Certain other restrictions and considerations for this entry will not be discussed here.

Column 17 applies only to combined files, primary input files, and secondary input files. It is used to indicate that the input or combined file named in columns 7 through 14 is to be checked for an end-of-file condition.

End of file means that there are no more cards to process because the end-of-file card (/* punched in columns 1 and 2) has been read. Whenever an end of file occurs, a last record (LR) indicator turns on. The job terminates after all operations conditioned by the LR indicator have been completed. The entry is especially significant when there are two input or combined files which do not contain the same number of cards. The entry in column 17 indicates which file causes an end of job.

Entry	*Explanation*
Blank	1. The job execution can end whether or not all of the records in this file have been processed. 2. If column 17 is blank for all of the input files, all records from every file must be processed before the program can end. The LR indicator turns on only after all files have reached end of file.
E	Check the file for end-of-file condition (input or combined file). If more than one file is to be checked, the LR indicator turns on only after all files to be checked reach the end of file.

Column 18 is used to describe the sequence of an input or combined file. In conjunction with other entries on the Input Specification sheet, the entry indicates which record fields are to be sequence checked, which fields are sequenced control fields, and which records are to be matched for processing. The following entries are made here:

Entry	*Explanation*
Blank	No sequencing or matching. Always blank for output, table, and demand files.
A	Ascending sequence.
D	Descending sequence.

Columns 19 through 23 may be left blank.

Columns 24 through 27 are used to indicate the length in characters of both input and output file records. The length must be within the size of the input or output device being used. For example, the record length for cards is always 96. The maximum length specified for the printer depends upon the number of printing positions available on the model being used: 96, 120, or 132. The entry must be right-justified.

Columns 28 through 31 are not used.

Column 32 may be used to assign two input or output areas to I/O files. When one I/O area is used, the computer reads, processes, and punches one card at a time. When two I/O areas are assigned, the computer reads or punches one card while processing another card record. The use of two areas can make the job execution faster. Since more storage space is required for the two records, the programmer must determine if this additional space will be available at execution time.

Entry	*Explanation*
Blank	One I/O area per file
1-9	Two I/O areas per file

Columns 33 and 34 are used to assign an overflow indicator to the printer. This indicator signals that the last line on a page has been printed. As a result, the correct action can be taken to eject to the next page, print headings, and so on. The functions of the indicator will not be discussed here.

Entry	*Explanation*
OA-OG or OV	Overflow indicator (any one symbol)

Columns 35 through 38 are not used.

Column 39 applies only to tables that are to be loaded at execution time and to output files that are to be printed on the line printer. The column is used in conjunction with the Extension Specification sheet to further describe the structure of the tables. The column is also used in conjunction with the Line Counter Specifications to further describe printed output.

Entry	*Explanation*
Blank	No line counter or extension specifications.
E	Extension specifications are used.
L	Line counter specifications are used.

Columns 40 through 46 are used to assign a specific input or output device to the file names in columns 7 through 14. Each device normally has only one file associated with it. The device code must start in column 40.

Entry	*Explanation*
MFCU1	Primary hopper of the MFCU.
MFCU2	Secondary hopper of the MFCU.
PRINTER	Printer. If the printer has a dual carriage feature, this entry refers to the left carriage.
PRINTER2	Right carriage of the printer, if the printer is equipped with the dual carriage feature.

Note: An output card file must first enter the System/3 by way of the MFCU, even though it may be blank cards before processing. The file is assigned to an MFCU stacker on the Output Format Specification sheet.

Columns 47 through 70 are not used.

Columns 71 and 72 are used to assign an external indicator to condition the file described on this line. The indicators control whether or not a file is to be used for a job at execution time.

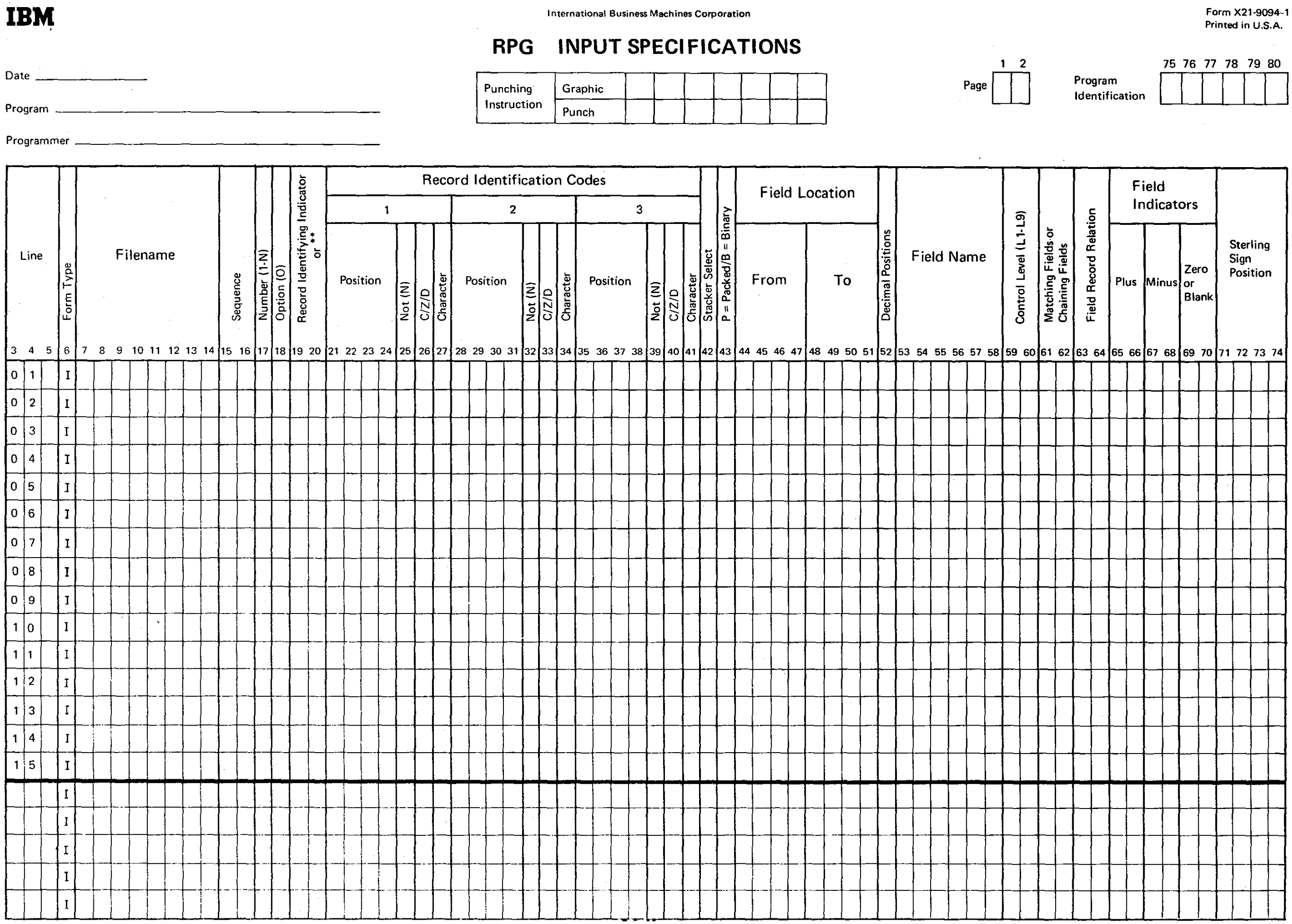
For example, for one run it may be required to produce both output cards and a printed report. For the next run, only the report may be needed. To avoid using two separate programs for the different runs, the external indicator can be set on or off according to the output required. If the indicator is on, the corresponding file is produced; if it is off, the file is not produced. Once the indicator has been set, it cannot be changed during an execution run. When the execution program is being loaded, provision is made to preset the indicators at that time.

External indicators may be used for all files: input, output, and combined files but not for table input files.

Entry	*Explanation*
U1 - U8	Indicators to condition a file

Input Specifications

Figure 20.4 is the RPG II Input Specifications form. It is used to further describe and identify the input files to a procedure. On this sheet the programmer may:



IBM

International Business Machines Corporation

Form X21-9094-1
Printed in U.S.A.

RPG INPUT SPECIFICATIONS

Date ____________

Program ____________

Programmer ____________

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes												Stacker Select	P = Packed/B = Binary	Field Location		Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators			Sterling Sign Position
							1				2				3																
							Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character			From	To						Plus	Minus	Zero or Blank	
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44 45 46 47	48 49 50 51	52	53 54 55 56 57 58	59 60	61 62	63 64	65 66	67 68	69 70	71 72 73 74
01	I																														
02	I																														
03	I																														
04	I																														
05	I																														
06	I																														
07	I																														
08	I																														
09	I																														
10	I																														
11	I																														
12	I																														
13	I																														
14	I																														
15	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 20.4 Input specifications.

1. Identify the different types of card records within a single file, for example, master name, address, transactions, and so on.

2. Specify the sequence, if any, of the various types of cards in a file.

3. Define card fields and identify the data by name in each field.

4. Set indicators based on individual fields.

5. Describe control fields.

6. Specify fields to be used for matching records or for sequence checking of data.

7. Direct input or combined file cards to specific stackers in the MFCU after processing.

The input specifications may be divided into two categories:

1. File and card type identification describe each input card and its relationship to other cards in the file.

2. Field description describes the individual fields in cards. These entries on the sheet must be written one line lower than file and card type identification entries.

Columns 1 through 7 and *75 through 80*. The uses of these columns on the sheet have been described in the previous section.

Columns 7 through 14 are used to identify the input file by name, the same file name previously assigned in the File Description Specifications.

Columns 15 and 16 are used to assign sequence numbers to different *types* of cards within an input file. For example, in a billing or invoice-writing procedure, the sequence of information on the printed form is predetermined. Normally, the customer name appears first, followed by street address, city and state, miscellaneous data, and transactions or items to be billed or invoiced. Transactions are listed in the body of the form after the heading information has been printed. The transactions may be in no particular sequence.

For this application, each type of card is assigned a sequence number beginning with the name as 01, street address as 02, and so on, with the transaction cards assigned the highest number in the sequence. Numbers from 01 to 99 may be assigned, but the sequence must begin with 01.

Figure 20.5 shows how the various types of input cards are arranged in the file for this job. A card out of sequence will cause the computer to halt at execution time for corrective action by the operator.

Entry	*Explanation*
01 - 99	Check for card sequence. No check is made on the sequence of data.
Any two alphabetic characters	Sequence of cards is not checked.

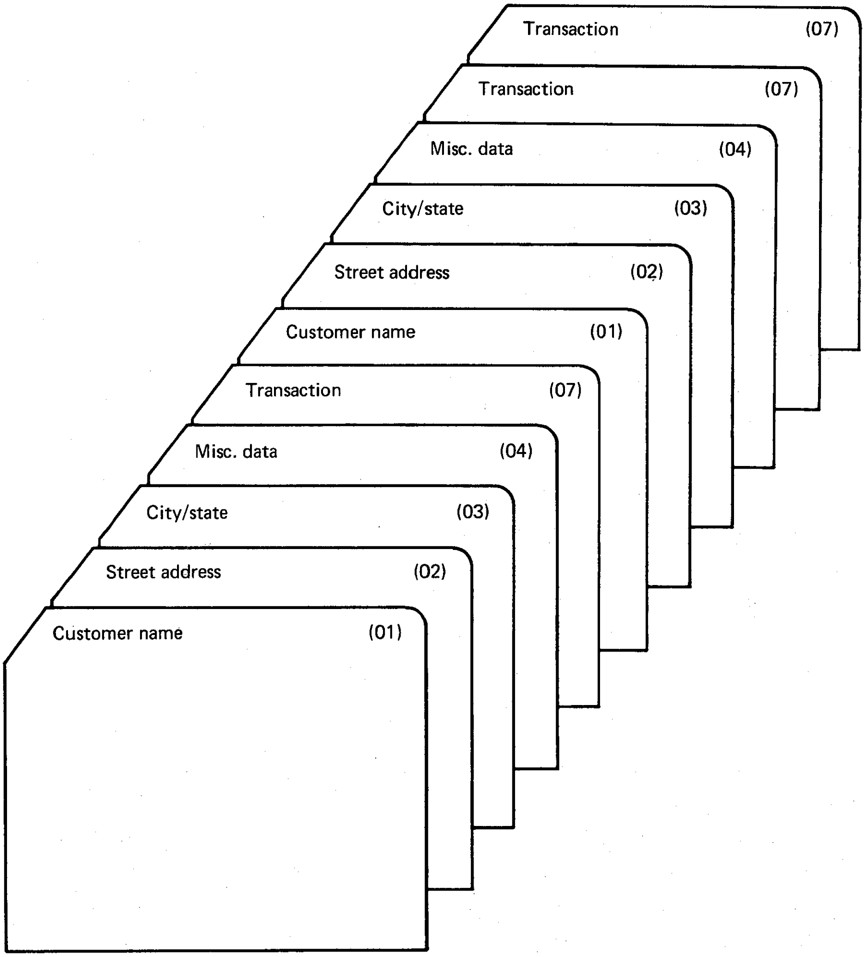


Figure 20.5 Card sequence. Invoice writing procedure.

Column 17 is used to indicate whether one card, or more than one card, of a particular type can be in the card sequence. An entry is made only if the corresponding numerical sequence entry is made in columns 16 and 17. Thus, in a billing operation the program can insure that no more than one customer name, address, or city/state card is in the file for each bill being printed. However, the program can be written to permit more than one transaction card per bill.

Entry	*Explanation*
Blank	Cards are not being checked for sequence. Columns 15 and 16 have alphabetic entries.
1	Only one card of this type may be present in the sequenced group.
N	One or more cards of a particular type may be present in the sequenced group.

If sequence checking entries are made in columns 15, 16, and 17, the types of cards checked *must* be in the file.

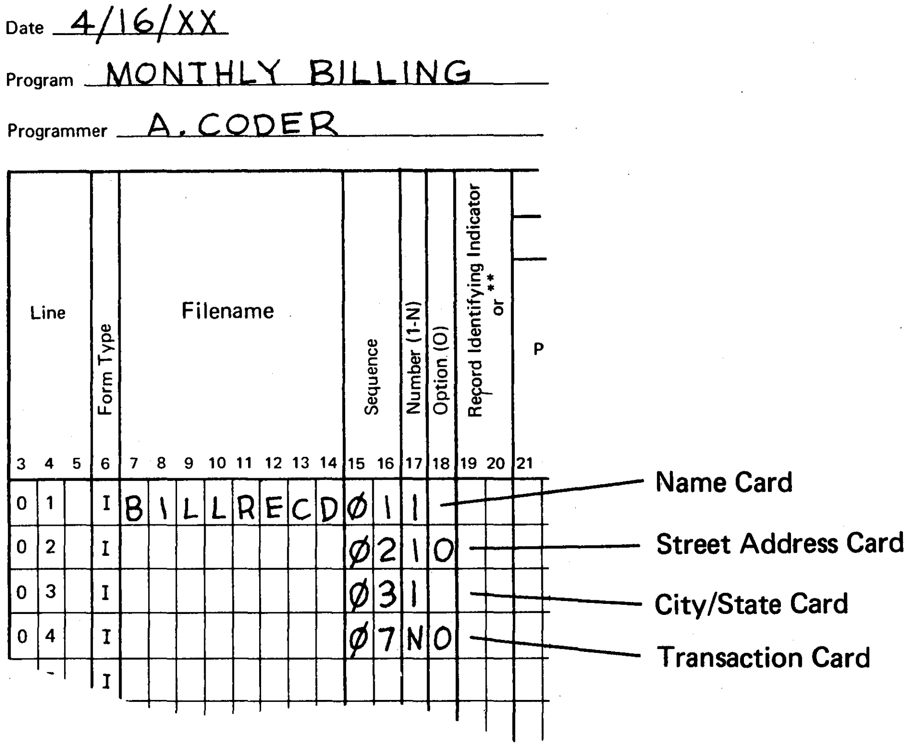
Column 18 makes this rule optional. An entry indicates that the card type being sequenced checked need *not* be in the file. For example, some invoiced customers may not have a street address. Therefore, a street address card does not have to be present for every customer.

Figure 20.6 shows RPG coding in columns 16 through 18 for the invoice example just discussed. The various types of cards are numbered 01, 02, 03, and 07. (Any two-digit ascending sequence may be used, starting with 01.) The file cards are sequence checked to be sure that each card is in its proper order for the invoice. Street address cards are considered optional.

In some instances, it might be required to run a report showing all customers, even though some customers have not made purchases during the period. For this procedure, item cards are also considered to be optional.

It is standard practice to construct unit-record files so that different types of cards are used for specific kinds of data. Each type of card is distinguished by unique hole punching in specified columns. For example, items purchased by a customer may be coded P in column 10 while items returned for credit may be coded C in the same column. The RPG system provides for recognition of card types.

Columns 19 and 20 are used to assign a record identifying indicator to each input record type.



Date 4/16/XX

Program MONTHLY BILLING

Programmer A. CODER

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	P	
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21	
0 1	I	BILLRECD	Ø1	1				Name Card
0 2	I		Ø2	1	O			Street Address Card
0 3	I		Ø3	1				City/State Card
0 4	I		Ø7	N	O			Transaction Card
-	I							

Figure 20.6 RPG coding. Invoice example.

It should be noted here that the indicators available to the program are not machine features. Program indicators are turned on or off by setting binary bits in predetermined locations of storage. Thus, program indicators may be used by the programmer in virtually the same manner as if they were designed into the computer.

When a specific indicator is assigned to a type of card record, the indicator is turned on as the card is selected for processing at execution time. The indicator specifies throughout the rest of the program cycle which card type has just been selected. As will be shown later, the indicators are used to condition calculation.

Columns 19 and 20 are also used to indicate that fields named in columns 53 through 58 of the following specifications are *look-ahead fields*. That is, a field in the next card to be processed may be examined to determine what operations should be done next. The operation is similar to using the first station reading brushes in an accounting machine to control the calculations to be done when a card reaches the next reading station.

The number of previous RPG entries have identified the various input files and records to the program. This identification must also be related to actual card punching of corresponding codes as it appears in the records.

Columns 21 through 41 are used for this purpose. No entries are required if all records are to be processed in the same manner. A maximum of three identifying characters may be shown on one specification line. Seven columns are provided for the description of one character or combination of punches in the code.

Columns 21 through 24, 28 through 31, and 36 through 38 specify the position of the character or code in the record.

Entry	*Explanation*
Blank	No record identification is needed.
01 - 96	Card column number of one character in the code.

Columns 25, 32, and 39 indicate whether the specified character *is* or *is not* in the record. The entry is equivalent to an X or No-X designation for punched-card accounting machines.

Entry	*Explanation*
Blank	Character is present in the code.
N	Character is not present in the code.

Columns 26, 33, and 40 indicate whether the entire character, only the zone portion, or only the digit portion is used in the identifying code.

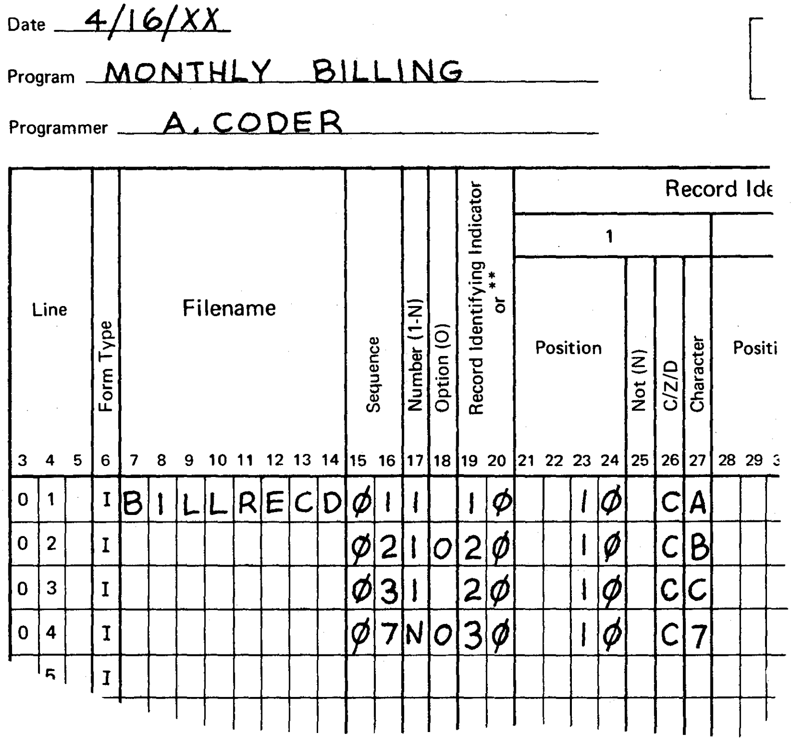
Entry	*Explanation*
C	Entire character
Z	Zone portion of the character
D	Digit portion of the character

When setting up record identifying codes, it is important to consider that many characters have the same zone or the same digit portion.

Columns 27, 34, and 41 identify the actual character. Any alphabetic character, special character, or digit can be used to serve as the code or part of the code.

Figure 20.7 shows four types of input records identified by the characters A, B, and C and the digit 7 in column 10. Figure

20.8 shows a payfile input record identified by the digit 6 in column 1, the digit 8 in column 90, and the zone portion of the character B in column 91.



Date 4/16/XX

Program MONTHLY BILLING

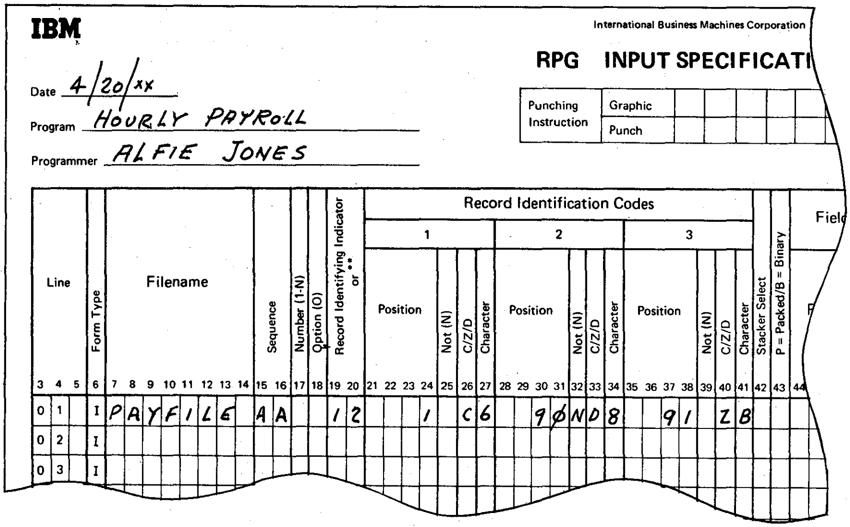
Programmer A. CODER

							Record Ide				
							1				
Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Position	Not (N)	C/Z/D	Character	Positi
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 3
0 1	I	BILLRECD	Ø1	1		1Ø	1Ø		C	A	
0 2	I		Ø2	1	O	2Ø	1Ø		C	B	
0 3	I		Ø3	1		2Ø	1Ø		C	C	
0 4	I		Ø7	N	O	3Ø	1Ø		C	7	
5	I										

Figure 20.7 Four types of input records.

A maximum of three identifying characters can be described on one specification line. If the identification code consists of more than three characters, an AND line must be used. This means that the first three identifying characters are described on one line. The additional identifying characters are described in as many of the following lines as are needed. The word AND is written in columns 14 through 16.

Figure 20.9 shows a payfile identified by four characters, using the AND line: digit 6 in column 10, digit 8 in column 90, zone portion of the character B in column 91, and the digit 5 in column 92.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATI

Date 4/20/xx

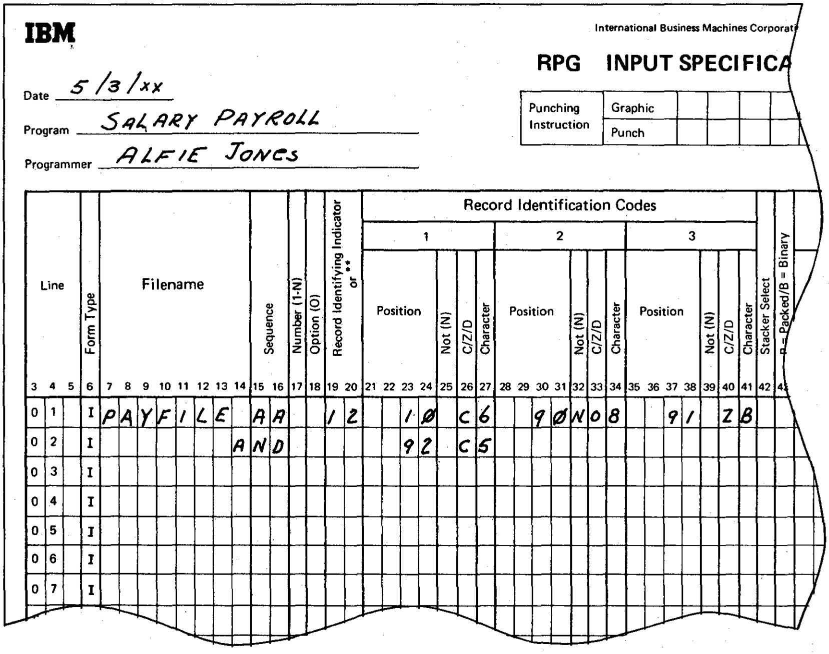
Program HOURLY PAYROLL

Programmer ALFIE JONES

Punching Instruction	Graphic
	Punch

Line			Form Type	Filename	Sequence		Number (1-N)	Option (O)	Record Identifying Indicator or **		Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	3 Position	Not (N)	C/Z/D	Character	Stacker Select	P = Packed/B = Binary	Fiel
3	4	5	6	7 8 9 10 11 12 13 14	15	16	17	18	19	20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44
0	1		I	PAYFILE	A	A			1	2	1		C	6	90	N	D	8	91		Z	B			
0	2		I																						
0	3		I																						

Figure 20.8 Payfile input record.



IBM

International Business Machines Corporati

RPG INPUT SPECIFICA

Date 5/3/xx

Program SALARY PAYROLL

Programmer ALFIE JONES

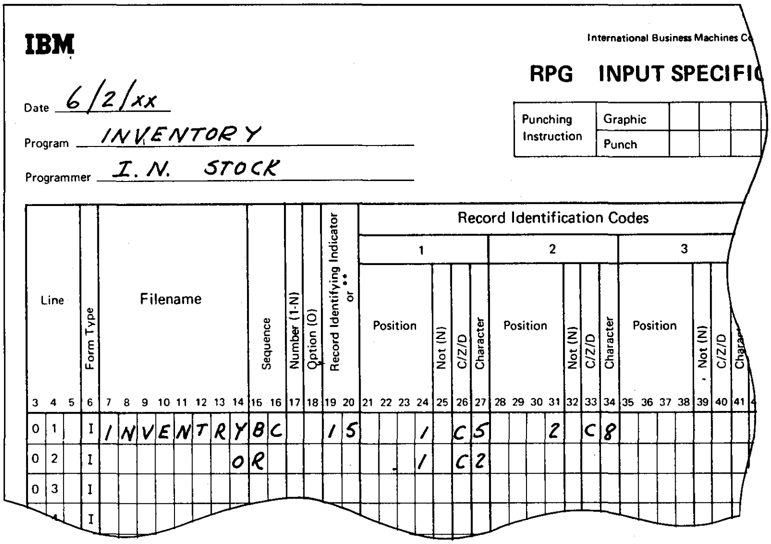
Punching Instruction	Graphic
	Punch

Line			Form Type	Filename	Sequence		Number (1-N)	Option (O)	Record Identifying Indicator or **		Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	3 Position	Not (N)	C/Z/D	Character	Stacker Select	P = Packed/B = Binary
3	4	5	6	7 8 9 10 11 12 13 14	15	16	17	18	19	20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43
0	1		I	PAYFILE	A	A			1	2	10		C	6	90	N	O	8	91		Z	B		
0	2		I	A	N	D					92		C	5										
0	3		I																					
0	4		I																					
0	5		I																					
0	6		I																					
0	7		I																					

Figure 20.9 Use of AND for record identification.

A particular card type may be identified by two different codes. In this case, an OR line can be used to indicate that either one of the codes may be present to identify the record. The word

OR is written in columns 14 and 15. Figure 20.10 shows a record identified with the digit 5 in column 1 and the digit 8 in column 2 OR the digit 2 in column 1.



IBM

International Business Machines C

RPG INPUT SPECIFI

Date 6/2/xx

Program INVENTORY

Programmer I. N. STOCK

Punching Instruction	Graphic			
	Punch			

							Record Identification Codes											
							1				2				3			
Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Charac
3 4 5	6	7 8 9 10 11 12 13 14	16 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41
0 1	I	INVENTRY	BC			15	1		C	5	2		C	8				
0 2	I	O	R				1		C	2								
0 3	I																	
	I																	

Figure 20.10 Use of OR for record identification.

Column 42 is used to specify an MFCU stacker (1-4) for the input cards after processing. If no entry is made, primary cards are automatically placed in stacker 1, the primary hopper. Secondary cards are placed in the secondary hopper, stacker 4. Combined files may be stacker-selected either on this form or on the Output Format Specification form.

If the same stacker is selected for both an input and an output file, a card from the output file is put into the stacker before a card from the input or combined file.

Entry	*Explanation*
Blank	Cards automatically fall into a primary or secondary stacker, 1 or 4.
1-4	Stacker where card type will be stacked.

Column 43 is not used.

Columns 44 through 48 describe the record fields of the input files. Columns 44 through 47 contain the number of the card column in which the field begins. Columns 48 through 51 contain the number of the card column in which the field ends. Both numbers (from 1-96) must be right-justified on the form and the leading zeros may be omitted. A single column field is defined by writing the same number in both the From and To entries.

Numerical fields may have a maximum length of 15 characters. Alphameric fields may be up to 256 characters long. However, fields read in from cards are restricted to the length of one punched card.

The number of positions to the right of the decimal is indicated for numerical fields in column 52. If there are no decimal positions, enter a zero.

Entry	*Explanation*
Blank	Alphameric field.
0-9	Number of decimal positions in a numerical field.

Input card fields are named in columns 53 through 58. The same name is used throughout the program to refer to the field. Only fields to be used in the processing need to be named. Rules:

1. Must begin with an alphabetic character.

2. May not contain special characters or blanks.

3. May be any combination of from one to six numbers and letters.

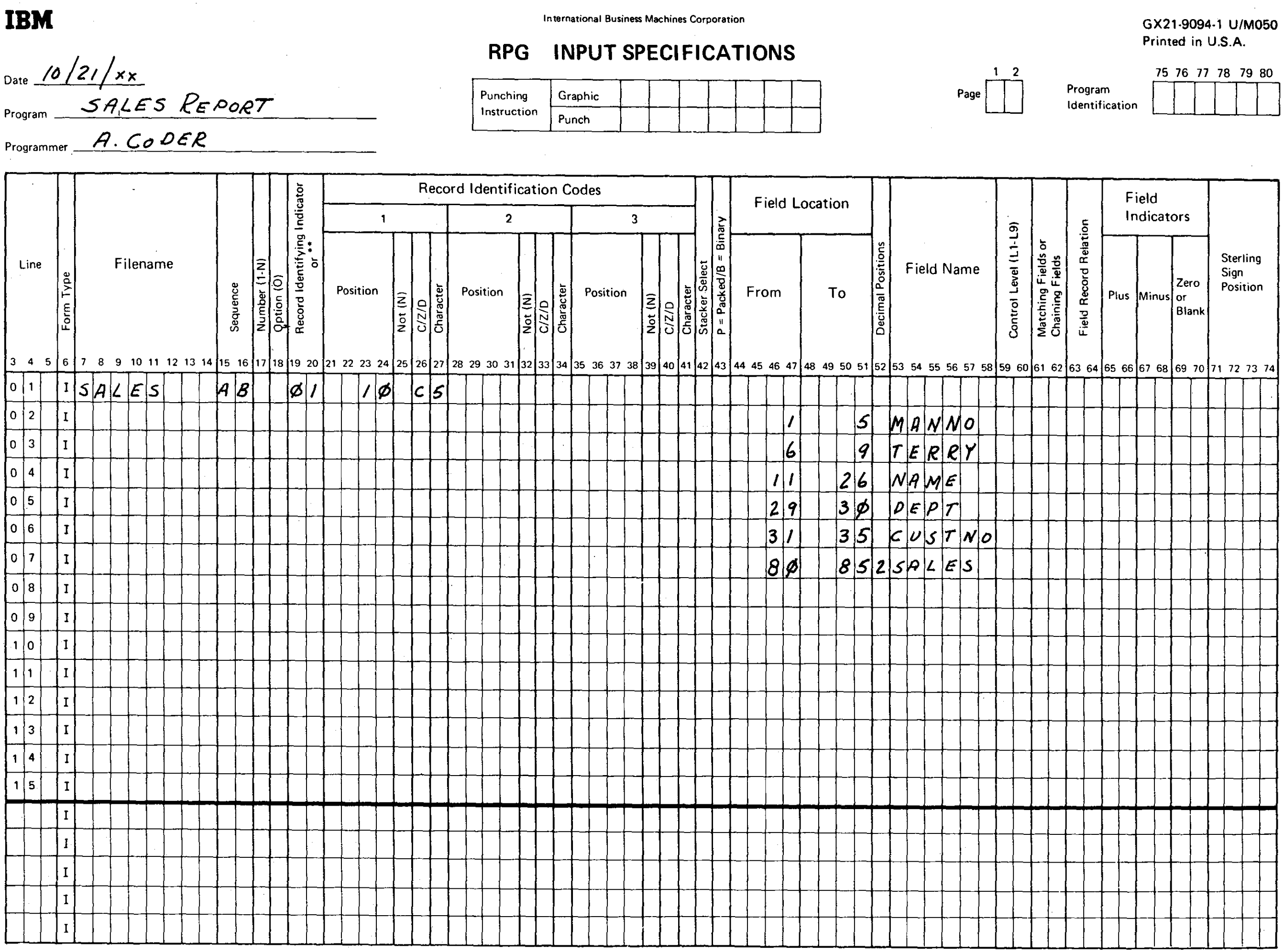
Figure 20.11 is an example of field description for a master salesman name and commission card.

If two or more card types contain identical fields, each field must be described. However, to avoid duplicate coding, an OR entry may be used. The OR relationship means that the fields named may be found in either one of the card types. OR lines can be used when –

1. Two or more card types have the same fields in the same positions.

2. Two or more card types have some fields which are the same and some fields which differ in location, length, or type of data.

The word OR is written in columns 14 and 15 to indicate an OR line as shown in Figure 20.12.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 10/21/xx

Program SALES REPORT

Programmer A. CODER

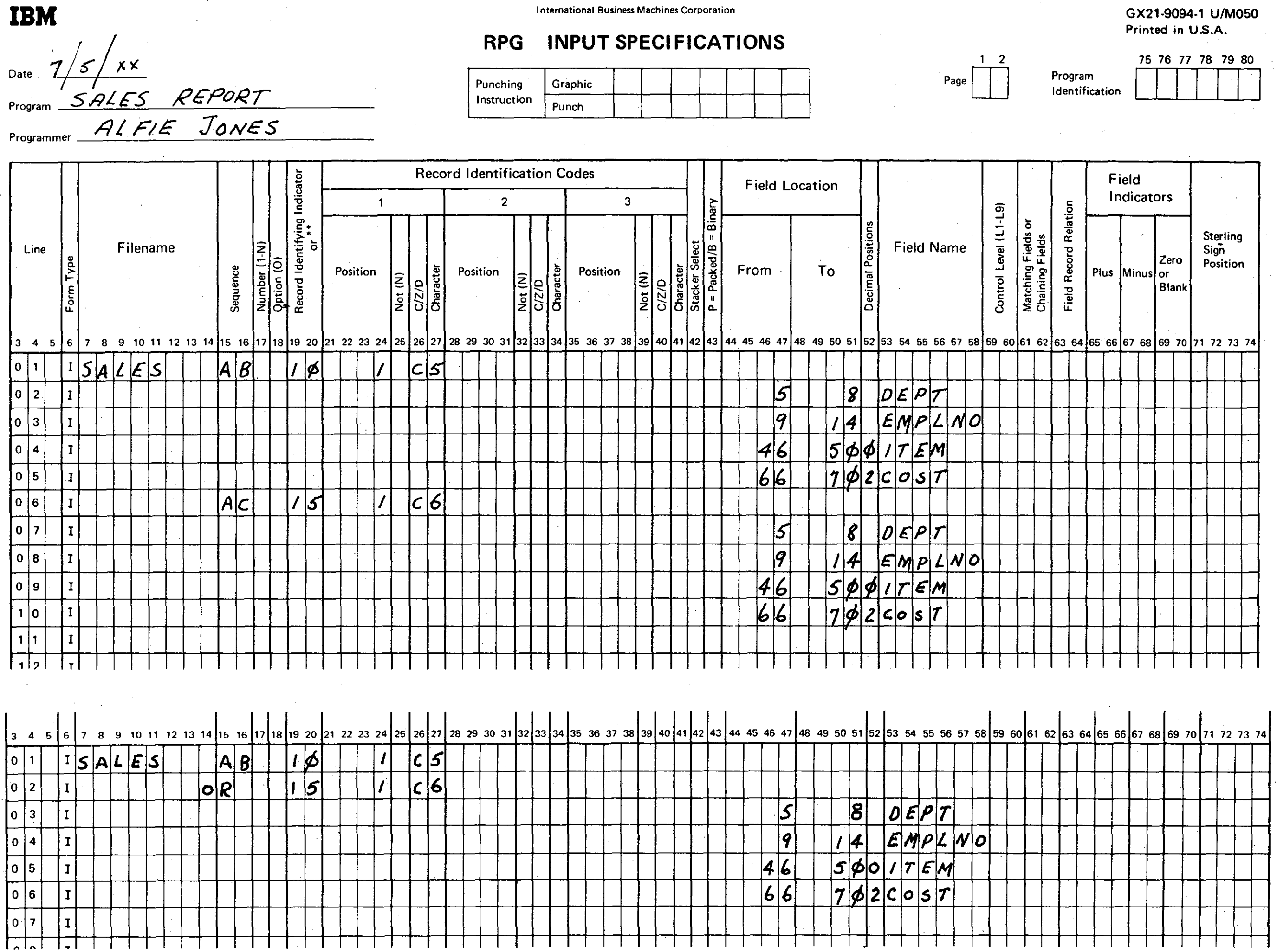
Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes												Stacker Select	P = Packed/B = Binary	Field Location		Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators			Sterling Sign Position
							1				2				3																
							Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character			From	To						Plus	Minus	Zero or Blank	
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44 45 46 47	48 49 50 51	52	53 54 55 56 57 58	59 60	61 62	63 64	65 66	67 68	69 70	71 72 73 74
01	I	SALES	AB			Ø1	1Ø		C	5																					
02	I																				1	5		MANNO							
03	I																				6	9		TERRY							
04	I																				11	26		NAME							
05	I																				29	3Ø		DEPT							
06	I																				31	35		CUSTNO							
07	I																				8Ø	85	2	SALES							
08	I																														
09	I																														
10	I																														
11	I																														
12	I																														
13	I																														
14	I																														
15	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 20.11 Field description.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 7/5/xx

Program SALES REPORT

Programmer ALFIE JONES

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	Record Identification Codes 2 Position	Not (N)	C/Z/D	Character	Record Identification Codes 3 Position	Not (N)	C/Z/D	Character	Stacker Select	P = Packed/B = Binary	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators Plus	Field Indicators Minus	Field Indicators Zero or Blank	Sterling Sign Position
01	I	SALES	AB			1ϕ	1		C	5																					
02	I																				5	8		DEPT							
03	I																				9	14		EMPLNO							
04	I																				46	5ϕ	ϕ	ITEM							
05	I																				66	7ϕ	2	COST							
06	I		AC			15	1		C	6																					
07	I																				5	8		DEPT							
08	I																				9	14		EMPLNO							
09	I																				46	5ϕ	ϕ	ITEM							
10	I																				66	7ϕ	2	COST							
11	I																														
12	I																														

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	Record Identification Codes 2 Position	Not (N)	C/Z/D	Character	Record Identification Codes 3 Position	Not (N)	C/Z/D	Character	Stacker Select	P = Packed/B = Binary	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators Plus	Field Indicators Minus	Field Indicators Zero or Blank	Sterling Sign Position
01	I	SALES	AB			1ϕ	1		C	5																					
02	I		OR			15	1		C	6																					
03	I																				5	8		DEPT							
04	I																				9	14		EMPLNO							
05	I																				46	5ϕ	O	ITEM							
06	I																				66	7ϕ	2	COST							
07	I																														

Figure 20.12 Field description. Two or more cards.

Field description entries can also be used for the page numbering of reports. These specifications will not be described here.

Columns 59 and 60 are used to assign control level indicators to specified fields of input records.

Card files normally enter the computer system in a predetermined sequence produced by mechanical sorting of control fields. When two or more types of cards are combined in the same sequence, groups of cards frequently contain like control numbers. The groups may represent such conditions as several purchases made by one customer, several job cards for the same employee, a number of inventory transactions for the same part number, and so on.

When the sequenced files are processed, amounts and quantities are accumulated and punched or printed by groups. With an accounting machine, this kind of operation is called group printing. As previously described, totals may be printed in minor, intermediate, or major groups depending upon the sequence of the files.

The RPG language provides for programming the System/3 to operate upon control fields in a manner similar to that of a punched-card accounting machine but with considerably more flexibility. For example, up to nine control levels may be programmed by assigning control level indicators to specified fields. Whenever a card with a control field is read, the field is compared with the same field in the previous card. The accounting machine compares cards at two different reading stations. The computer compares data in the control fields from two card records in storage.

When a control break occurs; that is, when a comparison between two fields is unequal, the corresponding indicator turns on. When the indicator is on, operations can be performed to cause total printing or punching as required. Normally control level indicators are used to –

1. Condition calculations to be performed whenever information in a control field changes.

2. Condition summary punching or total printing to be done after totals have been found for one control group.

3. Condition operations to be done on the card record that causes a change in a control field, for example, group indicate the next group.

Entry	*Explanation*
L1-L9	Any control level indicator.

Figure 20.13 shows entries to assign three control level indicators to the fields Employee Number, Department, and Location. L3 is the highest indicator and is equivalent to a major control in an accounting machine. When L3 is turned on, all lower level indicators are also turned on.

Additional rules, considerations, and restrictions for assigning control levels can be found in the IBM reference manuals for RPG II.

Columns 61 and 62 are used to compare records from two or more input and combined files. Records can be matched by one field, by as many as nine fields, or even by entire records. When match fields from a primary card are the same as match fields from a secondary card, the matching record (MR) indicator turns on.

Matched or unmatched records from two files are processed as follows:

1. When a card from the primary file matches a card from the secondary file, the primary card is processed first.

2. When cards from ascending files do not match, the card with the lower match field is processed first regardless of the file. Refer to Figure 19.14. When cards from descending files do not match, the card with the higher match field is processed first.

3. A card type with no matching field specification is processed immediately after the card it follows. The MR indicator is off. If this card type is first in the file, it is processed first even if it is not in the primary file as shown in Figure 20.14.

Entry	*Explanation*
M1-M9	Any matching level.

Rules:

1. Sequence checking is automatically done for all card types with matching field specifications. An error in sequence stops the program. All files containing records to be matched must be in the same sequence – either ascending or descending.

IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 1/15/xx

Program PAY LISTING

Programmer ALICE JONES

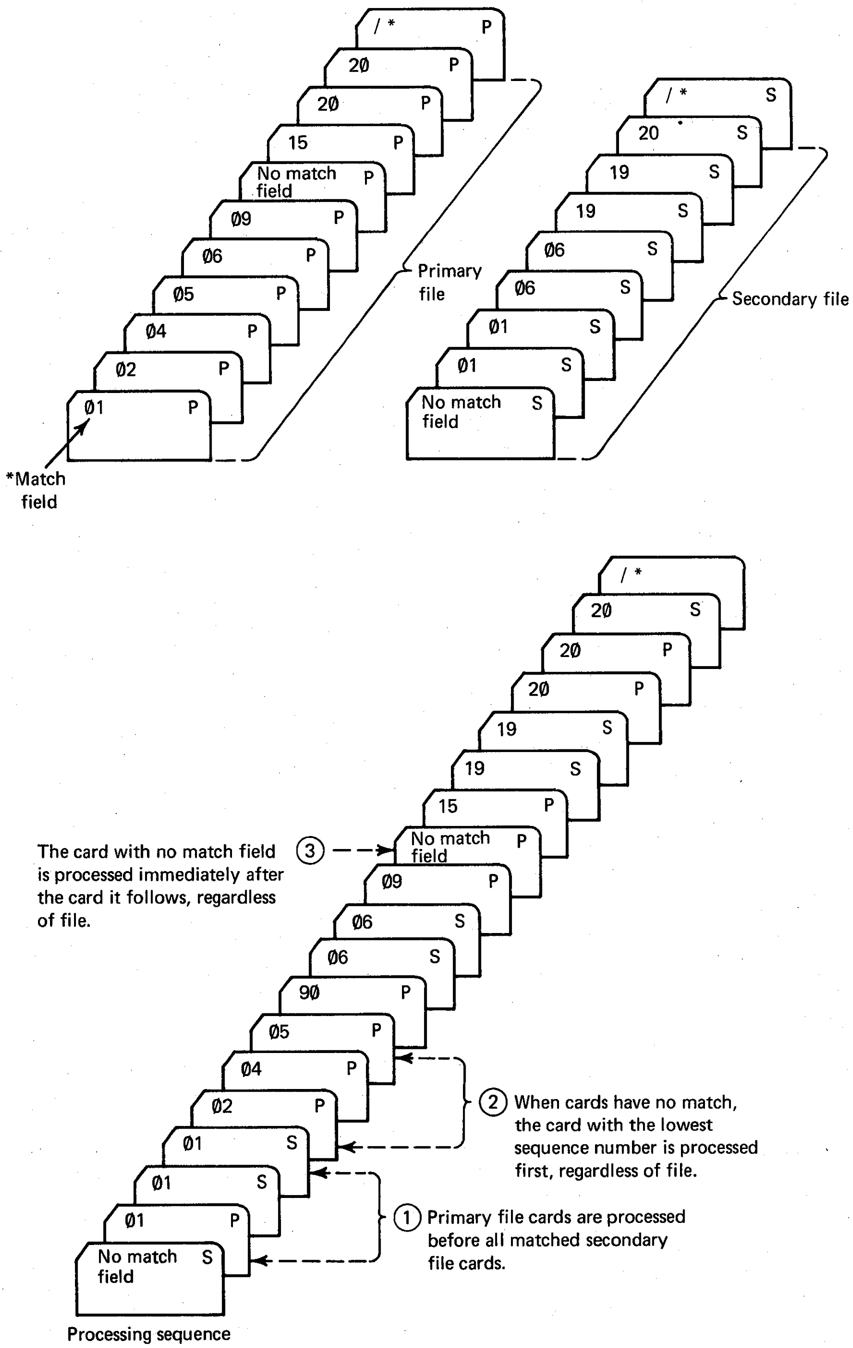
Punching Instruction	Graphic							
	Punch							

Page (1 2) □□

Program Identification (75 76 77 78 79 80) □□□□□□

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes												Stacker Select	P = Packed/B = Binary	Field Location		Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators			Sterling Sign Position
							1				2				3																
							Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character			From	To						Plus	Minus	Zero or Blank	
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44 45 46 47	48 49 50 51	52	53 54 55 56 57 58	59 60	61 62	63 64	65 66	67 68	69 70	71 72 73 74
01	I	PAYCARDS	AA			1Ø	1		C	A									2												
02	I																				5	1Ø		EMPLNO	L1						
03	I																				11	3Ø		NAME							
04	I																				8Ø	8Ø		DIVSON							
05	I																				33	33		SHIFT							
06	I																				2	4		DEPT	L2						
07	I																				78	79		LOCATN	L3						
08	I		CD			2Ø	1	N	C	A																					
09	I																				5	1Ø		EMPLNO	L1						
10	I																				11	14		DEPT	L2						
11	I																				6Ø	7Ø	1	HRSWKD							
12	I																				8Ø	8Ø		DIVSON							
13	I																				78	79		LOCATN	L3						
14	I																														
15	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 20.13 Control level indicators.



(3) The card with no match field is processed immediately after the card it follows, regardless of file.

(2) When cards have no match, the card with the lowest sequence number is processed first, regardless of file.

(1) Primary file cards are processed before all matched secondary file cards.

Figure 20.14 Processing order according to matching records.

2. At least one card type from each of the primary and secondary files must have matching fields.

3. The same number of matching fields must be specified for all card types used in matching. The same MR indicators must be used for all types.

4. All match fields given the same MR indicator must be the same length.

5. If more than one matching field is specified for a card type, all the fields are treated as one continuous matching field. They are combined according to ascending sequence of matching record indicators. The low field is on the right, the high field on the left, as shown below.

Division	*Department*	*Employee Number*
M3	M2	M1

6. Matching fields may be either alphamerical or numerical. However, all matching fields assigned a given level indicator (M1 through M9) are considered as numerical fields if any one of those fields is described as numerical.

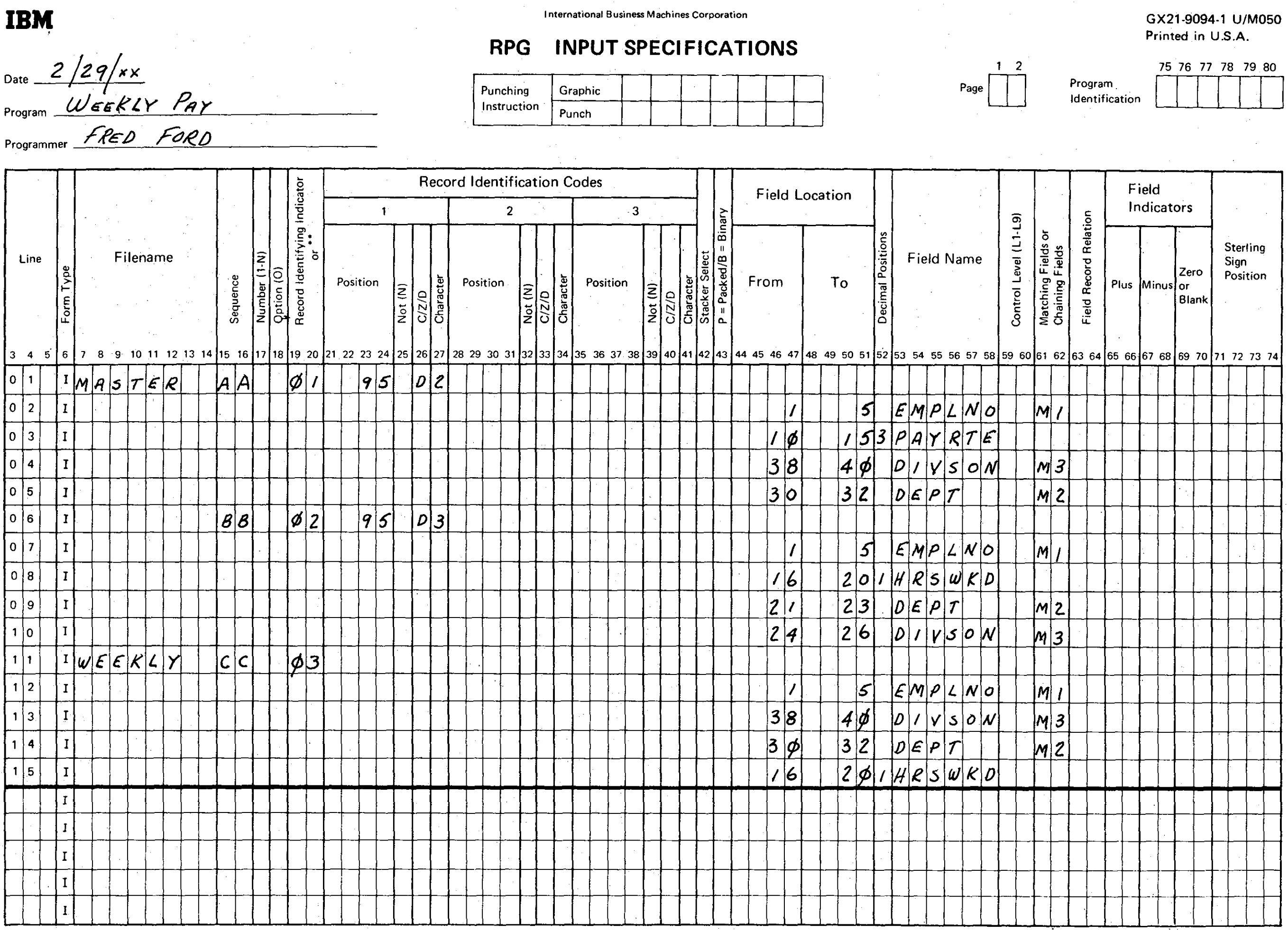
7. Only the digit portions of numerical fields are compared. Even though a field is negative, it is considered to be positive for comparing purposes. Thus, +123 will match −123.

8. Whenever more than one matching indicator is used, all match fields must match before the MR indicator turns on.

9. Field names are ignored in matching record operations. Therefore, fields from different card types which have the same match level may have different names.

Figure 20.15 is an example of three card types which are to be used in a matching operation. All card types have three matching fields specified and all use the same indicators (M1, M2, and M3) to indicate which fields must match. The MR indicator turns on only if all three match fields in either of the card types from the MASTER file are the same as all three fields from the card in the weekly file.

Figure 20.16 shows sequence checking for a single master card file.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 2/29/xx

Program WEEKLY PAY

Programmer FRED FORD

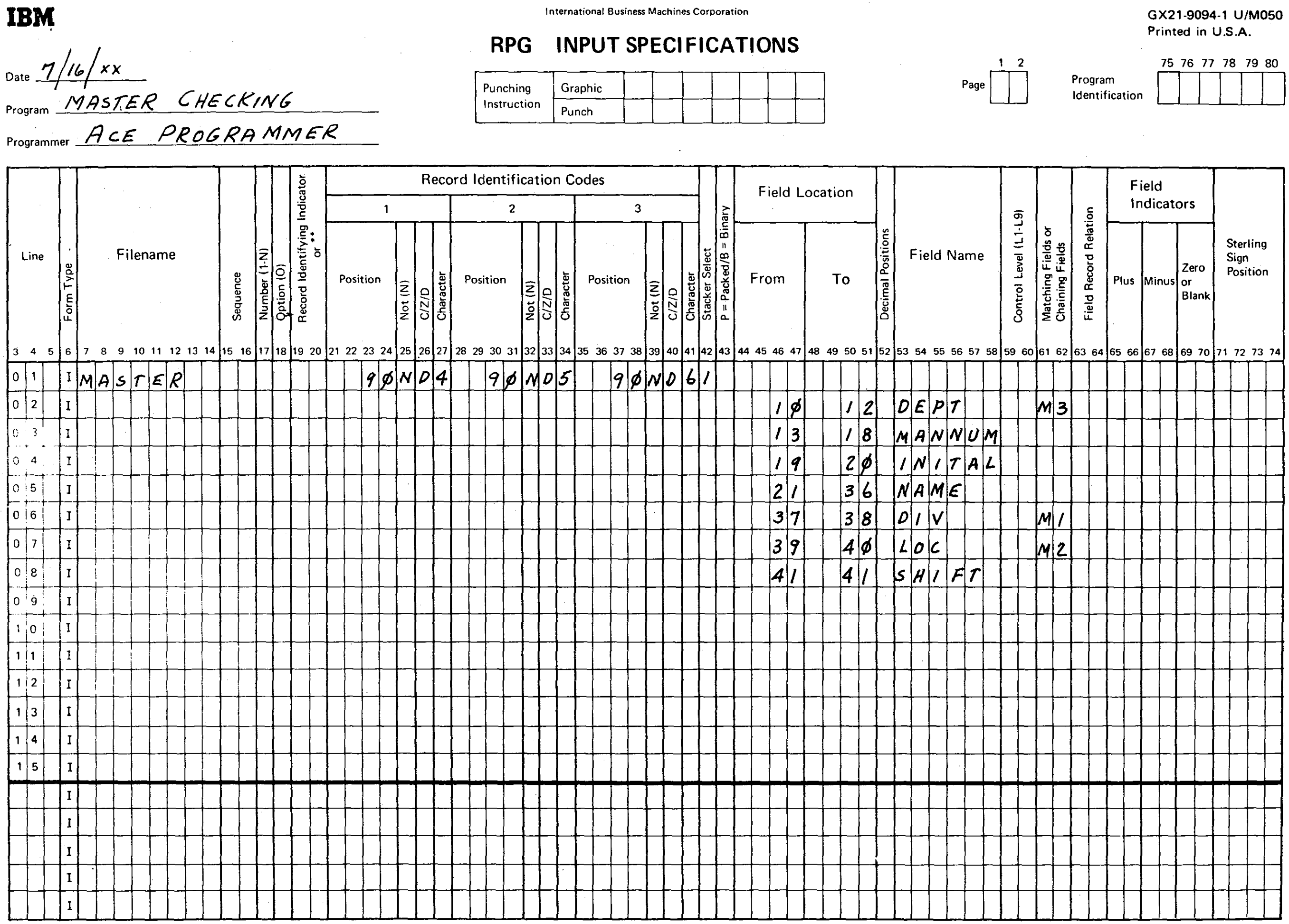
Punching Instruction	Graphic						
	Punch						

Page (1 2) □□

Program Identification (75 76 77 78 79 80) □□□□□□

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	P = Packed/B = Binary	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators Plus	Field Indicators Minus	Field Indicators Zero or Blank	Sterling Sign Position
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44 45 46 47	48 49 50 51	52	53 54 55 56 57 58	59 60	61 62	63 64	65 66	67 68	69 70	71 72 73 74
01	I	MASTER	AA			Ø1	95		D	2																					
02	I																				1	5		EMPLNO		M1					
03	I																				1Ø	15	3	PAYRTE							
04	I																				38	4Ø		DIVSON		M3					
05	I																				30	32		DEPT		M2					
06	I		BB			Ø2	95		D	3																					
07	I																				1	5		EMPLNO		M1					
08	I																				16	20	1	HRSWKD							
09	I																				21	23		DEPT		M2					
10	I																				24	26		DIVSON		M3					
11	I	WEEKLY	CC			Ø3																									
12	I																				1	5		EMPLNO		M1					
13	I																				38	4Ø		DIVSON		M3					
14	I																				3Ø	32		DEPT		M2					
15	I																				16	2Ø	1	HRSWKD							
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 20.15 Matching three card fields.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 7/16/xx

Program MASTER CHECKING

Programmer ACE PROGRAMMER

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

							Record Identification Codes														Field Location							Field Indicators			
							1				2				3																
Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Stacker Select	P = Packed/B = Binary	From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Plus	Minus	Zero or Blank	Sterling Sign Position
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44 45 46 47	48 49 50 51	52	53 54 55 56 57 58	59 60	61 62	63 64	65 66	67 68	69 70	71 72 73 74
01	I	MASTER					9Ø	N	D	4	9Ø	N	D	5	9Ø	N	D	6	1												
02	I																				1Ø	12		DEPT		M3					
03	I																				13	18		MANNUM							
04	I																				19	2Ø		INITAL							
05	I																				21	36		NAME							
06	I																				37	38		DIV		M1					
07	I																				39	4Ø		LOC		M2					
08	I																				41	41		SHIFT							
09	I																														
10	I																														
11	I																														
12	I																														
13	I																														
14	I																														
15	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 20.16 Sequence checking, master file.

Columns 63 and 64 are used for field record relation under three different situations:

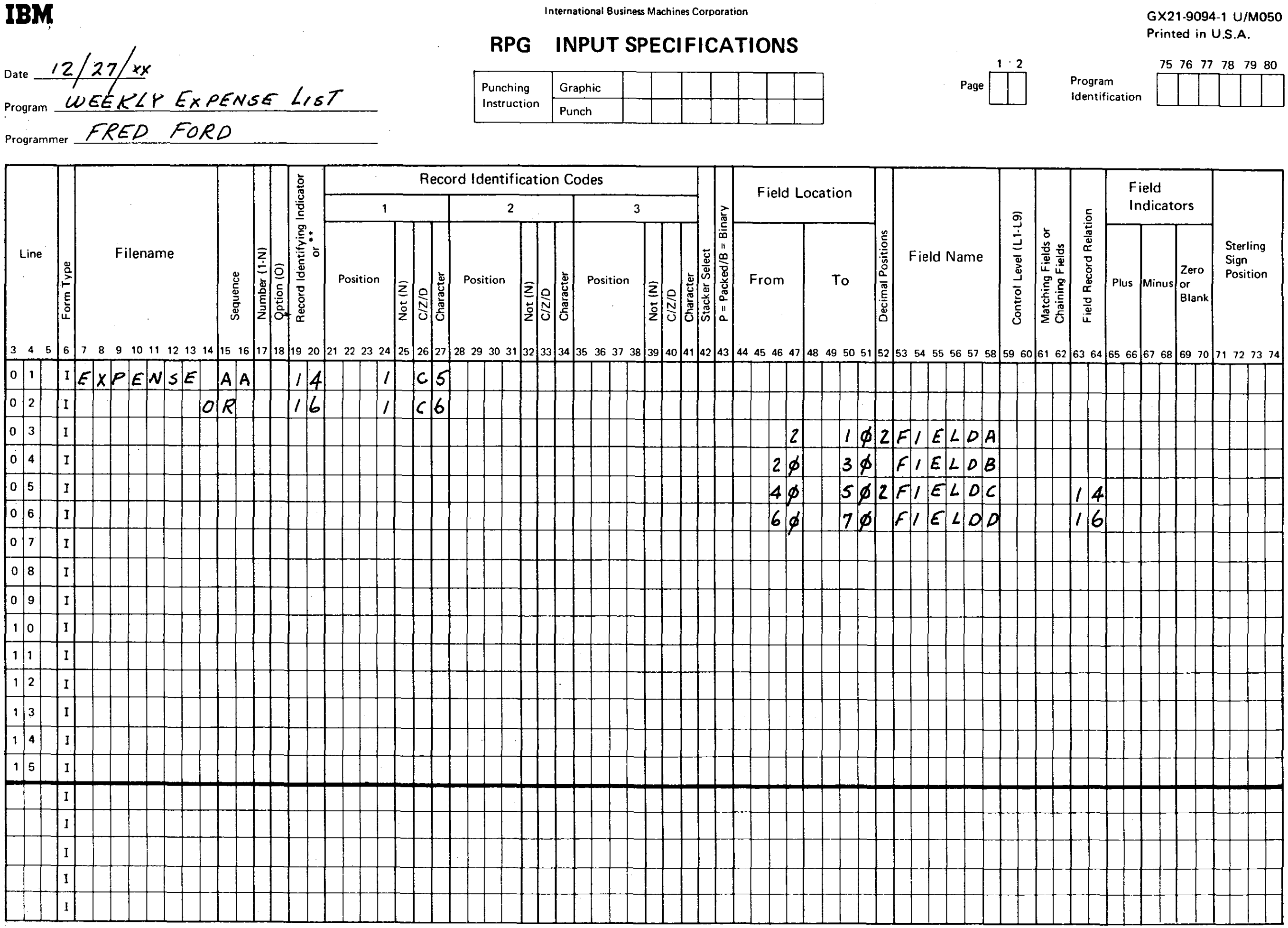
1. More than one card in an input file can be exactly related by entering an OR in columns 14 and 15. As shown in Figure 20.17, each record is assigned a record identifying indicator and entries are made for the unique card code in each card type. All card columns are commonly used in the specified OR relationship. That is, at execution time, cards with 5 punched in column 1 will be considered equally with cards punched with 6 in column 1.

However, it may be necessary to provide for exceptions to this rule, particularly when all fields in the two types of cards are *not* the same. Therefore, there must be some way to relate specific fields to particular OR cards as required. To do this, columns 63 and 64 should contain the same record identifying indicator shown in columns 19 and 20 of the card type where that field is located. In Figure 20.17, Field C is in columns 40 through 50 of the digit 5 card and Field D is in columns 60 through 70 of the digit 6 card. Field C is related to record indicator 14 and Field D is related to record indicator 16.

2. Data from an input card field may only be required when a control break or matching record occurs. In this case, columns 63 and 64 contain only the entry for the proper level indicators (L1-L9) or the MR indicator. An example is shown in Figure 20.18. The manager's name will be printed on the output report only when a control break in department occurs.

3. *Columns 63 and 64* may also be used to condition a specification by an external indicator (U1-U8). These indicators may be set prior to actual processing by a special control card used at system initialization time. When the indicator is on, the specification applies; when it is off, the specification is ignored.

Columns 65 through 70 are used to determine if the data content of an input record field is plus, minus, zero, or blank. From one to 99 program indicators are available. By entry in one of the field indicator spaces on the form, a specified indicator is associated with the field named in columns 53 through 58. At execution time, the indicator is turned on if a given condition is true and is turned off if the condition is not true. The indicators are used to condition either calculation or output operations.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 12/27/xx

Program WEEKLY EXPENSE LIST

Programmer FRED FORD

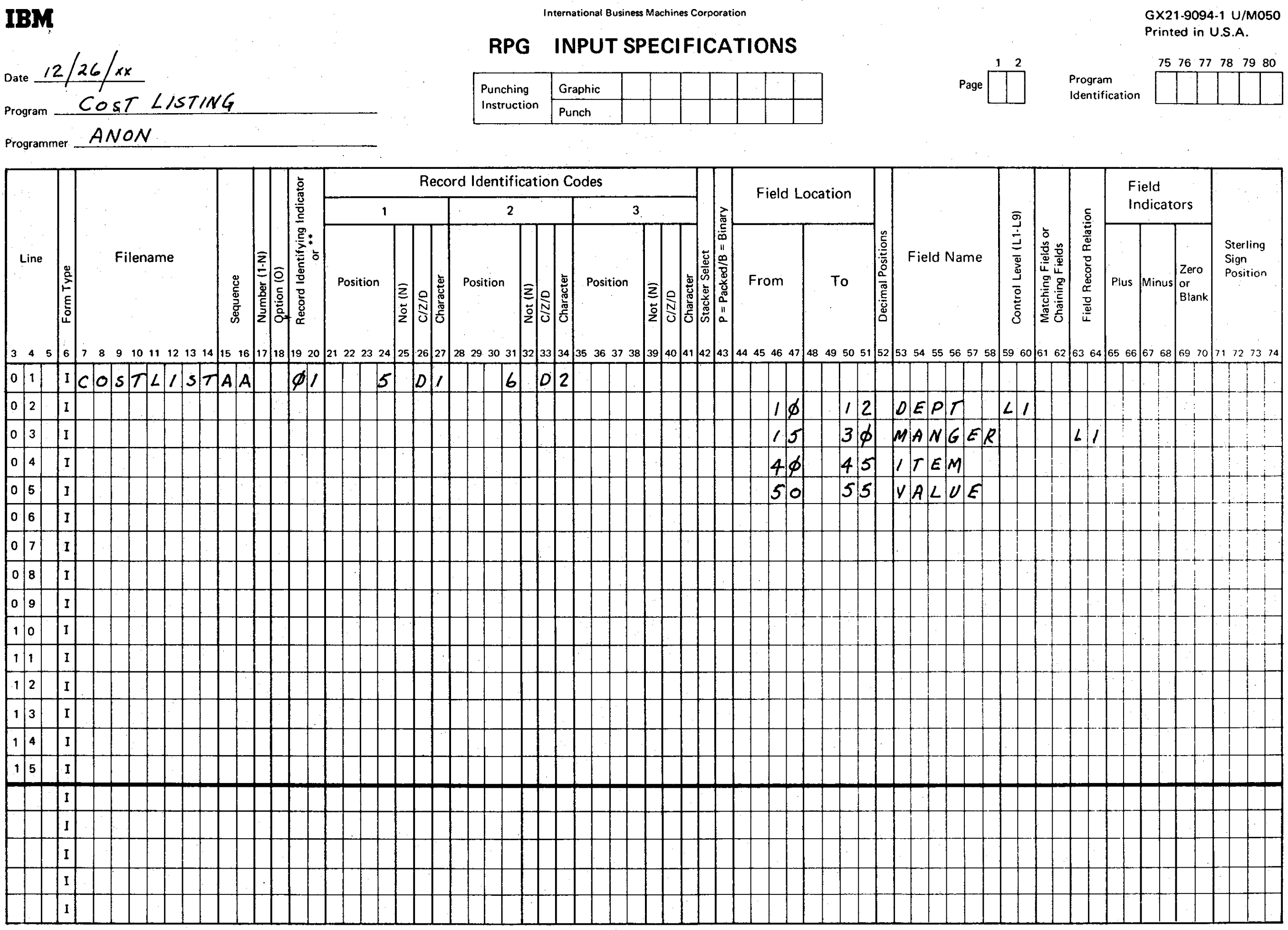
Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	3 Position	Not (N)	C/Z/D	Character	Stacker Select	P = Packed/B = Binary	Field Location From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators Plus	Minus	Zero or Blank	Sterling Sign Position
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44 45 46 47	48 49 50 51	52	53 54 55 56 57 58	59 60	61 62	63 64	65 66	67 68	69 70	71 72 73 74
0 1	I	EXPENSE	AA			14	1		C	5																					
0 2	I	O	R			16	1		C	6																					
0 3	I																				2	1Ø	2	FIELDA							
0 4	I																				2Ø	3Ø		FIELDB							
0 5	I																				4Ø	5Ø	2	FIELDC			14				
0 6	I																				6Ø	7Ø		FIELDD			16				
0 7	I																														
0 8	I																														
0 9	I																														
1 0	I																														
1 1	I																														
1 2	I																														
1 3	I																														
1 4	I																														
1 5	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 20.17 Field record relationship.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 12/26/xx

Program COST LISTING

Programmer ANON

Punching Instruction: Graphic / Punch

Page (1 2)

Program Identification (75 76 77 78 79 80)

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2: Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	3: Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	P = Packed/B = Binary (43)	Field Location From (44-47)	To (48-51)	Decimal Positions (52)	Field Name (53-58)	Control Level (L1-L9) (59-60)	Matching Fields or Chaining Fields (61-62)	Field Record Relation (63-64)	Field Indicators Plus (65-66)	Minus (67-68)	Zero or Blank (69-70)	Sterling Sign Position (71-74)
01	I	COSTLIST	AA			Ø1	5		D	1	6		D	2																	
02	I																				1Ø	12		DEPT	L1						
03	I																				15	3Ø		MANGER			L1				
04	I																				4Ø	45		ITEM							
05	I																				50	55		VALUE							
06	I																														
07	I																														
08	I																														
09	I																														
10	I																														
11	I																														
12	I																														
13	I																														
14	I																														
15	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 20.18 Printing after matching records.

Entry	*Explanation*
01-99	Field indicators.
H1-H9	Halt indicators.

The following conditions may be checked:

1. Plus (columns 65 and 66). Any indicator entered here (from 01-99) turns on when the numerical field named in columns 53 through 58 is greater than zero.

2. Minus (columns 67 and 68). Any indicator entered here (from 01-99) turns on when the numerical field named in columns 53 through 58 is less than zero.

Plus or minus indicators are off at the beginning of execution time. They are not turned on until a plus or minus condition is satisfied by the field being tested on the record just read. Plus or minus indicators are not valid for alphameric fields.

3. Zero or blank (columns 69 and 70). Any indicator entered here turns on when the field named in columns 53 through 58 is either all zeros or all blanks. A numerical field is tested for zeros; an alphabetic field is tested for blanks.

An indicator assigned to zero or blank is off at the beginning of execution time. It remains off until the field being tested is not zero or blank in the record just read.

Halt indicators (H1-H9) may also be specified in any of the three field indicator columns. The indicators are normally used to recognize error conditions in input records that should not be processed, for example, a minus net pay in a payroll run. The indicator entered is associated with the field named in columns 53 through 58.

If halt indicators are turned on, execution stops immediately before the next record is read. Total calculations conditioned by control level indicators (see columns 7 and 8 under calculation specifications) and total output operations are not performed before the halt. The halt indicator is turned off immediately before the halt actually occurs.

Questions and Exercises

1. What steps are required to produce an RPG object program deck?
2. What kinds of files are handled by RPG?
3. What is the purpose of the control or header specification?
4. Describe any circumstances you can think of where a change in collating sequence might be necessary.
5. What is a combined file?
6. What is the purpose of the input specifications?

Case Study Problem

Figure 20.19 is a system diagram of an inventory stock status run on the System/3 Model 10. The diagram shows that two input files are fed into the primary and secondary hoppers of the MFCU.

Primary File; four types of cards:

Pricing masters	type code 1
Requisitions	” ” 2
On order	” ” 3
Stock status	” ” 4

The primary file is in ascending numerical sequence by a four-digit item number. Cards are also in type code sequence within item, as shown in the diagram. The following table lists the data fields as punched in each type of card.

		Card Type Code			
Field	*Columns*	1	2	3	4
Month and day	1 - 4	x	x	x	x
Type code	5	x	x	x	x
Item number	6 - 9	x	x	x	x
Quantity	10 - 14		x	x	x
Unit of measure	15 - 17	x	x	x	x
Description	18 - 32	x			x
Unit price	33 - 37	x			x
Inventory value	38 - 43				x
Department using	44 - 45		x		
Authorized by	46 - 55		x	x	
Old stock balance	65 - 69				x
Total withdrawals	70 - 74				x
New stock balance	75 - 79				x
Total on order	80 - 84				x
Total available	85 - 89				x

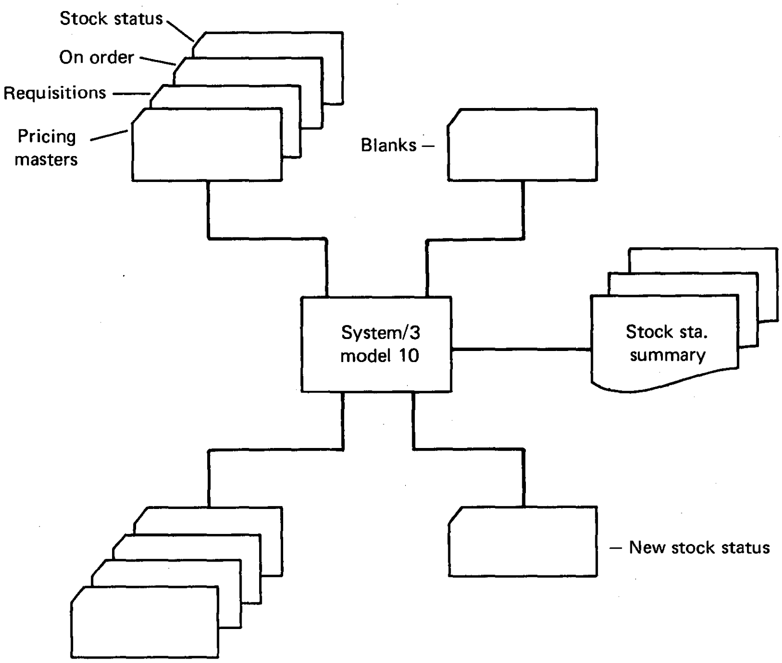


Figure 20.19

Secondary – blank stock status cards.

Output files are as follows:

Pocket 1 – file from primary feed of MFCU.

Pocket 3 – new stock status summary cards (type code 4) with quantity and amount fields updated with new balances. These cards will be used for the next period run as the "old balance" file.

Printer – Stock status summary showing all fields produced for the summary card. The report is to be group printed in item sequence.

7. Prepare RPG II control or header card, file description, and input specifications for the above run.

Note: The specifications of the various files are intended as guides for programming the problem. The student may arrange his data differently to suit his own interpretation of the result sought. For example, only the inventory value is to be calculated, according to the procedure. However, in an actual situation, it would probably be practical to also compute value of stock withdrawn and on order. These amounts can be punched into the requisition and on-order cards; the cards can be separated from the file after the stock status run, and a printed report prepared.

21 RPG II CALCULATION

After all input files have been described, the next step in preparation of the source program is that of specifying what calculations are to be performed. Figure 21.1 shows the form to be used for this purpose. Entries describe three aspects of the various operations that can be programmed in the RPG language.

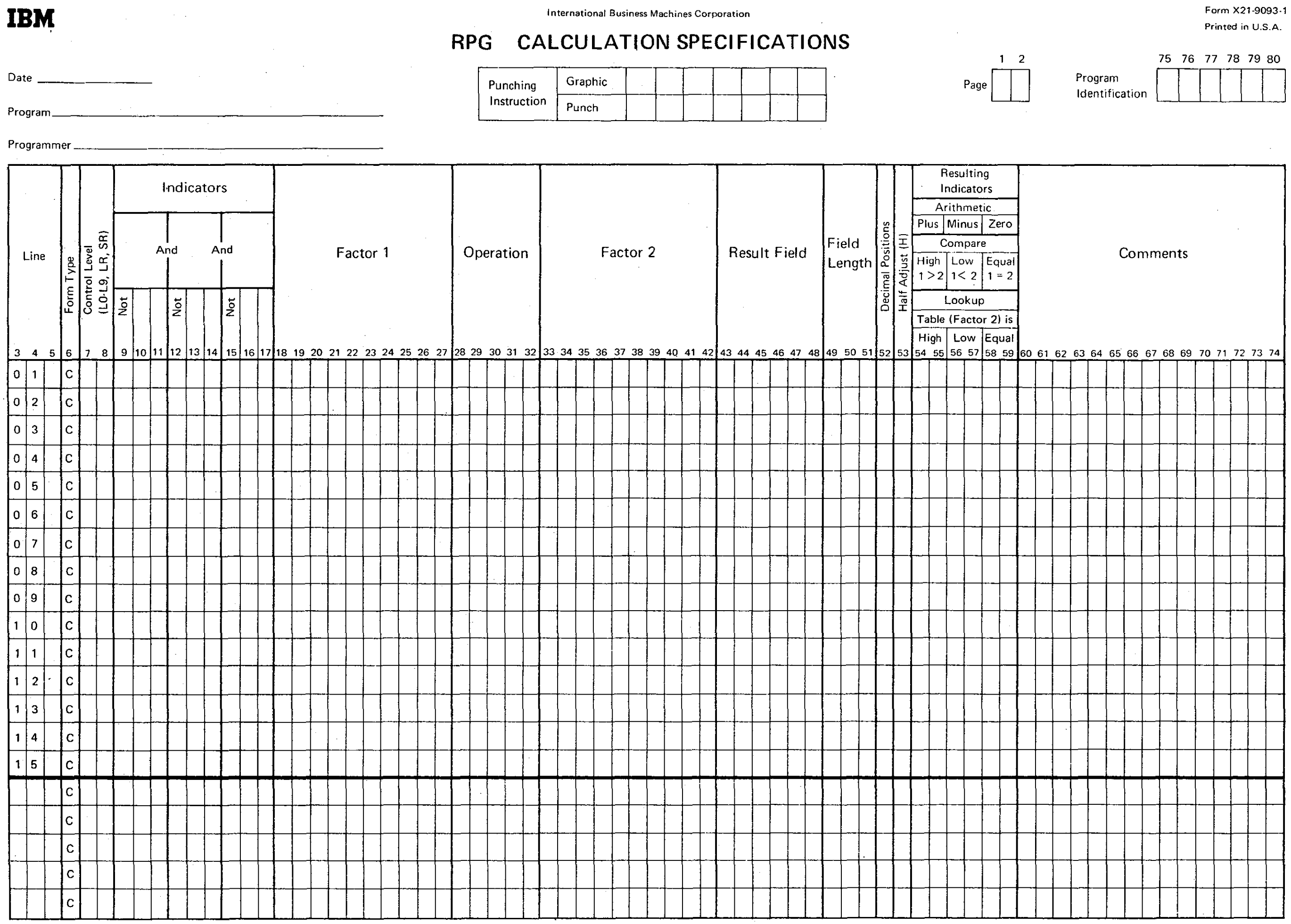
1. When an operation is to be performed. Indicators entered in columns 7 through 17 determine under what conditions a calculation is to be carried out.

2. What operation is to be performed: add, subtract, move data, compare, half adjust, and so on. Operations are specified in the following format:

FACTOR 1   OPERATION   FACTOR 2   RESULT

These entries, written in columns 18 through 48, describe the four parameters involved in all calculation. They are placed on one line of the specification sheet. Field length, decimal positions, and half adjust are also entered on the same line.

3. What tests are to be made on the calculation results. Result indicators in columns 54 through 59 signal the condition



IBM

International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

Form X21-9093-1
Printed in U.S.A.

Date ____________

Program ________________________________

Programmer ________________________________

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Control Level (L0-L9, LR, SR)	Indicators						Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators			Comments
					And		And									Arithmetic			
			Not		Not		Not									Plus	Minus	Zero	
																Compare			
																High 1 > 2	Low 1 < 2	Equal 1 = 2	
																Lookup			
																Table (Factor 2) is			
																High	Low	Equal	
3 4 5	6	7 8	9	10 11	12	13 14	15	16 17	18 19 20 21 22 23 24 25 26 27	28 29 30 31 32	33 34 35 36 37 38 39 40 41 42	43 44 45 46 47 48	49 50 51	52	53	54 55	56 57	58 59	60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
0 1	C																		
0 2	C																		
0 3	C																		
0 4	C																		
0 5	C																		
0 6	C																		
0 7	C																		
0 8	C																		
0 9	C																		
1 0	C																		
1 1	C																		
1 2	C																		
1 3	C																		
1 4	C																		
1 5	C																		
	C																		
	C																		
	C																		
	C																		
	C																		

Figure 21.1 Calculation specifications.

of the results of an operation and may be set to condition following operations.

All operations are performed in the sequence entered unless otherwise specified.

Columns 1 through 7. The use of these columns has been previously explained.

Columns 7 and 8 are used to –

1. condition an operation with a control level indicator;
2. specify that the item on this line is part of a subroutine.

Entry	*Explanation*
Blank	The operation specified on this line is performed for every card read – provided the indicators in columns 9 through 17 are properly set.
LO	The LO indicator is on during the entire program execution time. It is used when no control fields have been assigned and when it is required to perform total calculations and to write and punch total output records. Written lines conditioned by LO should appear on the form before lines conditioned by L1-L9 or LR.
L1-L9	Control level indicators. The operation described on this line is performed only when the specified control indicator is on. The indicator turns on when data changes in a control field. The original assignment of these indicators is made in columns 59 and 60 of the Input Specifications form. A control break in one level causes all lower level indicators to turn on. Operations conditioned by these indicators will also be performed.
LR	Last record indicator turns on when the last card has been read and processed. (All other indicators specified [L1-L9] are also turned on.) The operation specified on this line will be performed only after the last record has been read, for example, print a final total for the run.
SR	The operation on this line is part of a subroutine. This function will be explained later.

Columns 9 through 17 are used to assign indicators which control when an operation is or is not to be performed. Three separate fields are available on the form: columns 9 through 11, columns 12 through 14, and columns 15 through 17. From one to three indicators may be used to condition an operation. If the indicator must *not* be on to condition the operation, an N is written in column 9, 12, or 15.

All three indicators are in an AND relationship with each other. That is, the status of all indicators must be exactly as specified before the operation is performed. The indicators are also in an AND relationship with the control level indicators specified in columns 7 and 8.

Entry	*Explanation*
Blank	Operation is performed for every card read.
01-99	Field or resulting indicators used elsewhere in the program. Refer to columns 65 through 70 of the Input Specification form.
L1-L9	Control level indicators previously assigned.
LR	Last record indicator.
MR	Matching record indicator.
H1-H9	Halt indicators assigned elsewhere.
U1-U8	External indicators previously set.
OA-OG, OV	Overflow indicator previously assigned.

The following are additional considerations for the assignment of the various indicators:

1. Record identifying indicators previously specified in columns 19 and 20 on the Input Specifications form can condition an operation for execution only for a certain type of card.

2. Field indicators previously specified in columns 65 through 70 on the input form can condition an operation for execution after the status of a field has been checked and has met certain conditions.

3. Resulting indicators entered in columns 54 through 59 of this form can condition an operation according to the results of previous calculations.

4. Halt indicators previously used in columns 65 through 70 on the input form or in columns 54 through 59 on this form prevent an operation from being carried out when a certain error condition has been found in the input data or in the result of a calculation. Specification of the halt indicator is necessary because the card that caused the halt condition will be completely processed before the execution of the object program stops.

A halt indicator may also be used to condition an operation to be done *only* when a specific error occurs.

5. The MR indicator conditions an operation to be done only when matching records have been found.

6. External indicators condition which operations should be done and which files used for a specific run. Indicators may have been previously specified on the File Description form in columns 71 and 72.

7. The LR indicator conditions all operations to be done at the end of a run.

8. Control level indicators used in these columns and previously specified in columns 59 and 60 on the input form condition the operation to be performed on only the first card of a new control group.

9. Overflow indicators previously specified in columns 33 and 34 on the File Description form condition operations to be performed when the last printed line on a page has been reached.

Note that the above conditioning of data processing operations is very similar to the functions performed by pilot and digit selectors, coselectors, and other devices in an accounting machine. The RPG language provides the same ability to calculate from selected categories of cards in a file and to make logical decisions based on conditions encountered at execution time.

Columns 18 through 27 are used to enter FACTOR 1.

Columns 33 through 42 are used to enter FACTOR 2.

The following entries can be used:

1. The name of any record field that has been defined.

2. Any alphameric or numerical constant.

3. Any subroutine, table, or array name.

4. Any date field names (UDATE, UMONTH, UDAY, UYEAR).

5. The special name PAGE.

The entries used depend upon the operation being performed. Items 3, 4, and 5 will not be explained in this text.

Constants

A constant is the actual data to be used at execution time rather than a name or symbolic representation of the data. In the RPG system, the constant may be either alphameric or numerical. Entry of the constant is left-justified in either FACTOR 1 or FACTOR 2.

The following rules apply to *numerical constants*:

1. The constant is any combination of the digits 0-9. A decimal point and a minus sign may be included in the field, as required. Examples: 1234, 0100, 0.0100, –123, –.0123.

2. An unsigned constant is assumed to be plus. If present, the minus sign must be the leftmost character.

3. Blanks may not appear in the constant, thus: 12b34, .b001, –b.09.

4. Numerical constants are used in the same way as numerical fields in specific records.

5. The maximum length of the constant is 10 positions, including the sign and decimal point.

The following rules apply to *alphameric constants*:

1. Any combination of characters can be used, including blanks. Examples: 'DECEMBER,' 'DEC,' 'DECb25,' '12b25bXX.'

2. Alphameric constants must be enclosed by apostrophes as shown above.

3. Alphameric constants may not be used for arithmetic operations.

4. The maximum length of an alphameric constant is 10 characters, including the apostrophes.

Columns 28 through 32 are used to specify the operation to be performed using FACTOR 1, FACTOR 2, and the result field. The RPG language has a special set of operation codes that can be performed. Each code requires certain entries on the same specification line. These rules of "grammar" are summarized in Figure 21.2 and all available codes are listed. Each operation will be briefly described in the next section. The following rules apply:

1. The operation code must begin in column 28.

Type of Operation		Operation Code (columns 28-32)	Control Level	Indicators	Factor 1	Factor 2	Result Field	Field Length	Decimal Position	Half Adjust	Resulting Indicators
Arithmetic Operations	Add Factor 2 to Factor 1.	ADD	0	0	R	R	R	0	0	0	0
	Clear Result Field and add Factor 2.	Z–ADD	0	0	B	R	R	0	0	0	0
	Subtract Factor 2 from Factor 1.	SUB	0	0	R	R	R	0	0	0	0
	Clear Result Field and subtract Factor 2.	Z–SUB	0	0	B	R	R	0	0	0	0
	Multiply Factor 1 by Factor 2.	MULT	0	0	R	R	R	0	0	0	0
	Divide Factor 1 by Factor 2.	DIV	0	0	R	R	R	0	0	0	0
	Move remainder of preceding division to a Result Field.	MVR	0	0	B	B	R	0	0	R	0
	Sum elements of the array and put sum in Result Field.	X FOOT	0	0	B	R	R	0	0	0	0
Move Operation	Move Factor 2 into Result Field, right justified.	MOVE	0	0	B	R	R	0	0	B	B
	Move Factor 2 into Result Field, left justified.	MOVEL	0	0	B	R	R	0	0	B	B
	Move zone from low-order position of Factor 2 to low-order position of Result Field.	MLLZO	0	0	B	R	R	0	0	B	B
	Move zone from high-order position of alphameric Factor 2 to high-order of alphameric Result Field.	MHHZO	0	0	B	R	R	0	B	B	B
	Move zone from low-order position of Factor 2 to high-order position of alphameric Result Field.	MLHZO	0	0	B	R	R	0	B	B	B
	Move zone from high-order position of alphameric Factor 2 to low-order position of Result Field.	MHLZO	0	0	B	R	R	0	0	B	B
Compare and zone testing operations	Compare Factor 1 to Factor 2.	COMP	0	0	R	R	B	B	B	B	R
	Identify the zone in the leftmost position of an alphameric Result Field.	TESTZ	0	0	B	B	R	B	B	B	R
Setting Indicators	Set one, two, or three specific indicators on.	SETON	0	0	B	B	B	B	B	B	R
	Set one, two, or three specific indicators off.	SETOF	0	0	B	B	B	B	B	B	R
Branching within RPG 11	Branch to another RPG 11 calculation specification line.	GOTO	0	0	B	R	B	B	B	B	B
	Identify the name in Factor 1 as a destination label to which GOTO may branch.	TAG	0	B	R	B	B	B	B	B	B
Lookup Operations	Table Lookup.	LOKUP	0	0	R	R	0	0	0	B	R
	Array Lookup.	LOKUP	0	0	R	R	0	0	0	B	R
Subroutine	Beginning of the subroutine.	BEGSR	B	B	R	B	B	B	B	B	B
	End of the subroutine.	ENDSR	B	B	0	B	B	B	B	B	B
	Call to execute the subroutine.	EXSR	0	0	B	R	B	B	B	B	B
Program Control	Forcing record to be read next.	FORCE	B	0	B	R	B	B	B	B	B
	Forcing output printing.	EXCPT	0	0	B	B	B	B	B	B	B
Debug Function	Aid in finding programming errors.	DEBUG	0	0	0	R	0	B	B	B	B

0 - Optional
R - Required
B - Blank

Figure 21.2 RPG II operation summary.

2. All operations are performed in the order specified on the calculation form.

3. All operations conditioned by control level indicators in columns 7 and 8, except those which are part of a subroutine, must follow those that are not conditioned by control level indicators.

Columns 43 through 48 are used to name the field which will contain the result of the operation specified in columns 28 through 32. A new field may be named, either alphameric or numerical. The following rules apply:

1. The result field must begin with an alphabetic character and contain no blanks or special characters.

2. If the field has not been described elsewhere, columns 49 through 52 should contain an entry to define the length of the new field.

3. The field may also be described elsewhere, but it must be described in exactly the same manner.

Columns 49 through 51 are used to designate the length of a result field. Numerical fields may have a maximum length of 15 digits. Alphameric fields may be up to 256 characters long. If the designated length is not sufficient to hold the result, digits or characters will be dropped.

Column 52 indicates the number of positions to the right of the decimal point in a numerical field. If the result is alphameric, the column is left blank. If the numerical field has no decimal positions, enter a zero.

The number of decimal positions can never be greater than the length of the field; the maximum number of positions is 9.

However, the number of positions may be more than the number of decimal positions that actually result from an operation. In this case, zeros are filled in to the right. If the number of decimal positions is less than the actual number that results from an operation, the extra positions to the right are dropped.

Column 53 is used to indicate that the contents of the result field are to be half adjusted or rounded.

Entry	*Explanation*
Blank	Do not half adjust.
H	Half adjust.

The adjustment is made by adding 5 to the first digit to the right of the last decimal position specified in column 52. If the field is negative, a minus 5 is added. All decimal positions to the right of the number specified in column 52 are dropped.

Example:	123.4679	Product of a multiply operation.
	5	Add 5 to the digit to the right of the last decimal position specified.
	123.5XXX	Drop all decimal positions beyond the number specified in column 52.
	123.5	Result after half adjusting.

Column 53 should be left blank for all nonarithmetic operations.

Columns 54 through 59 are used to specify indicators that are turned on by the result of the operation specified in columns 28 through 32. Following operations can then be conditioned by testing these indicators.

Three fields are provided for result indicators: columns 54 and 55, 56 and 57, and 58 and 59. Each field is used for different conditions: plus or high, minus or low, and zero or equal.

Indicators may also be turned on or off by the special operations SETON and SETOF. Thus, the condition of an indicator can be under control of the programmer. Normally, only indicators 01-99 and H1-H9 are used in these fields, but other types of indicators can be used if required.

Entry	*Explanation*
01-99	Any two-digit number.
H1-H9	Any halt indicator.
L1-L9	Any control level indicator.
LR	Last record indicator.
OA-OG, OV	Any overflow indicator.

Columns 54 and 55 (plus or high): An indicator specified in this field is turned on when:

1. The result field in an arithmetic operation is plus.

2. FACTOR 1 is higher than FACTOR 2 in a compare operation.

3. FACTOR 2 is higher than FACTOR 1 in a table look-up operation.

4. The zone tested in a TESTZ operation is a plus zone.

Columns 56 and 57 (minus or low). An indicator specified in this field turns on when:

1. The result field in an arithmetic field is minus.

2. FACTOR 1 is lower than FACTOR 2 in a compare operation.

3. FACTOR 2 is lower than FACTOR 1 in a table look-up operation.

4. The zone tested in a TESTZ operation is a minus zone.

Columns 58 and 59 (zero or equal). An indicator specified in this field turns on when:

1. The result field in an arithmetic operation is zero.

2. FACTOR 1 is equal to FACTOR 2 in a compare operation.

3. FACTOR 2 is equal to FACTOR 1 in a table look-up operation.

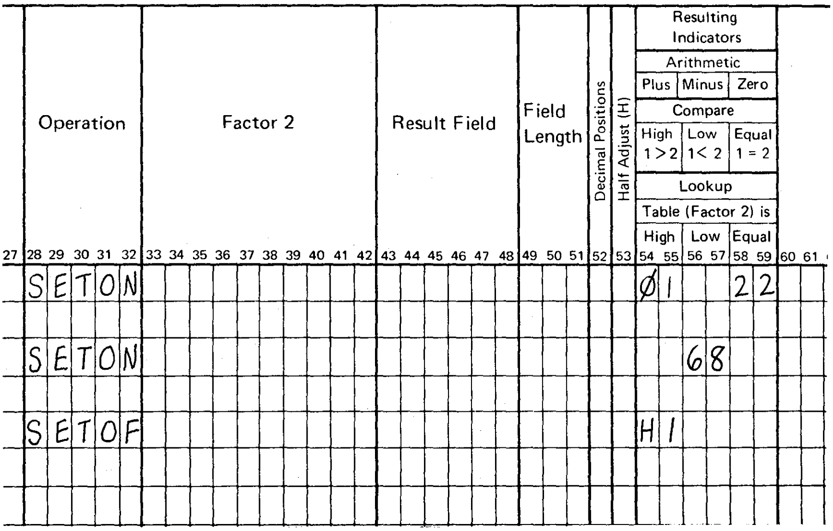
4. The zone tested in a TESTZ operation is neither plus or minus.

The following rules also apply to result indicators:

1. When the same indicator is used to test the result of more than one operation, the operation last performed determines the setting of the indicator.

2. Any indicators to be turned on or off by the SETON or SETOF operations can be entered in any of the three fields (columns 54 through 59, as shown in Figure 21.3.

3. Halt indicators are normally used only to check for error conditions.



	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Arithmetic Plus Compare High 1>2 Lookup Table (Factor 2) is High	 Minus Low 1< 2 Low	 Zero Equal 1 = 2 Equal	
27	28 29 30 31 32	33 34 35 36 37 38 39 40 41 42	43 44 45 46 47 48	49 50 51	52	53	54 55	56 57	58 59	60 61
	SETON						Ø1		22	
	SETON							68		
	SETOF						H1			

Figure 21.3 Result indicators.

Columns 60 through 74 on the Calculation Specifications sheet are used for comments.

Use of Result Indicators

Figure 21.4 is a program calculation routine that illustrates the use of the result indicators just described. The routine calculates gross pay by employee in a payroll application. The routine is also flowcharted in the illustration to show the logic of the procedure.

The item numbers below refer to line numbers on the Calculation Specifications sheet.

01. Compare the employee's hours worked to the constant 40. (Forty is the standard number of hours in the work week.) Set indicator 10 on if hours worked are greater than 40 and set indicator 20 on if hours worked are less than 40. If hours worked equal 40, no indicator is set. As a result of this comparison and the setting of the indicators, the machine will follow one of three possible paths shown as A, B, and C on the flowchart.

Path A: Hours worked less than 40.

02. Indicator 10 is off; this calculation is not executed.

03. Indicator 20 is on. This calculation is executed only when indicator 20 is *not* on.

04, 05. Indicator 10 is off; these steps are ignored.

06. Indicator 20 is on; pay rate is multiplied by actual hours worked to obtain gross pay.

Path B: Hours worked greater than 40.

02. Indicator 10 is on; the constant 40 is subtracted from the actual hours worked to obtain overtime hours.

03. Indicator 20 is not on; the pay rate is multiplied by 40 to obtain regular pay.

04. Indicator 10 is on; overtime rate is multiplied by overtime hours to obtain overtime pay.

05. Indicator 10 is on; overtime pay is added to regular pay to obtain total pay.

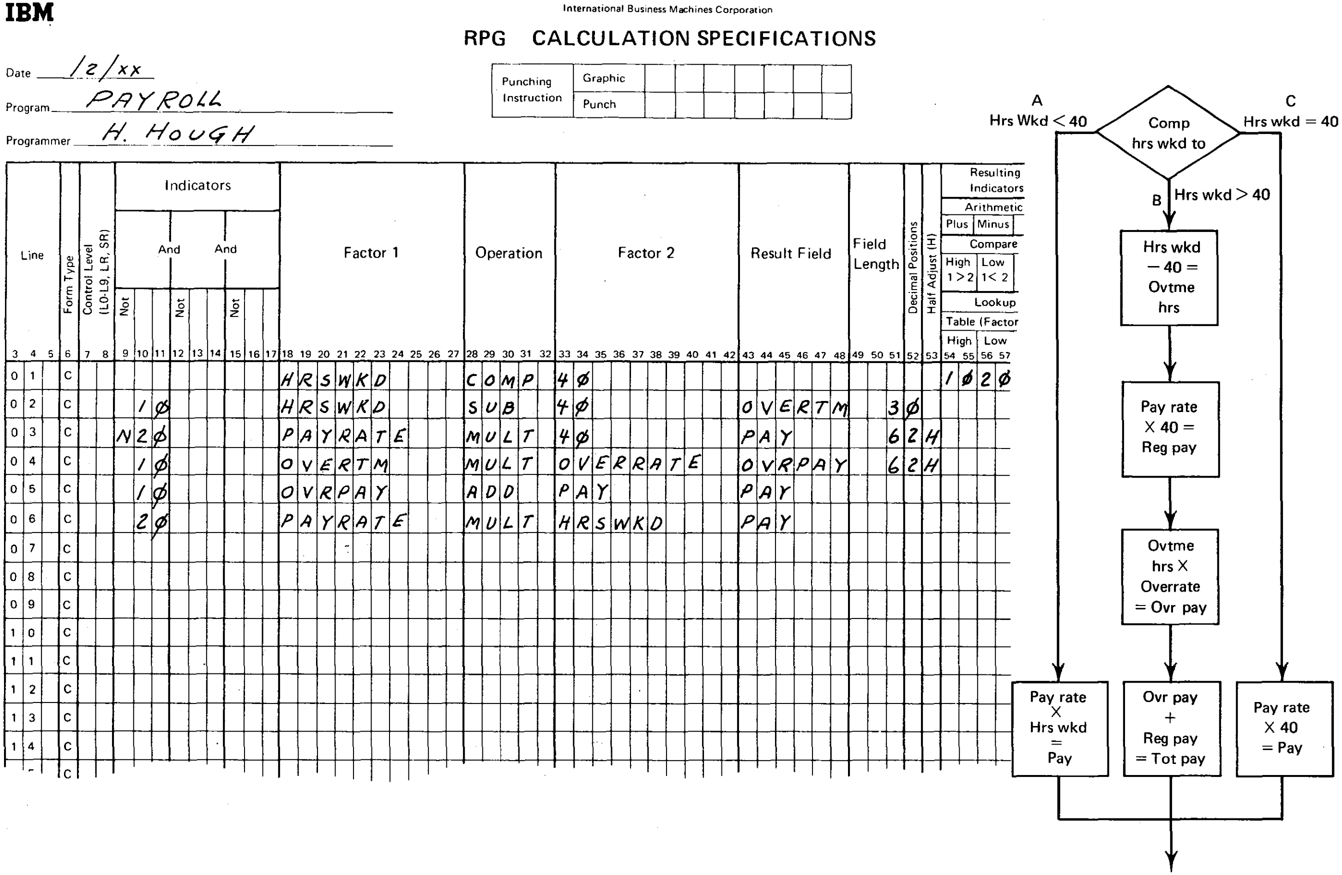
06. Indicator 20 is not on; this step is ignored.

Path C: Hours worked equal 40.

02. Indicator 10 is not on; this step is ignored.

03. Indicator 20 is not on; pay rate is multiplied by 40 to obtain gross pay.

04, 05, 06. Both indicators are off; these steps are ignored.



IBM

International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

Date /2/xx

Program PAYROLL

Programmer H. HOUGH

Punching Instruction	Graphic							
	Punch							

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR) (7-8)	Indicators: Not (9-11)	And: Not (12-14)	And: Not (15-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Arithmetic Plus / Compare High 1>2 / Lookup (Factor 2) is High (54-55)	Minus / Low 1<2 / Low (56-57)
01	C					HRSWKD	COMP	4Ø					1Ø	2Ø
02	C		1Ø			HRSWKD	SUB	4Ø	OVERTM	3	Ø			
03	C		N2Ø			PAYRATE	MULT	4Ø	PAY	6	2	H		
04	C		1Ø			OVERTM	MULT	OVERRATE	OVRPAY	6	2	H		
05	C		1Ø			OVRPAY	ADD	PAY	PAY					
06	C		2Ø			PAYRATE	MULT	HRSWKD	PAY					
07	C													
08	C													
09	C													
10	C													
11	C													
12	C													
13	C													
14	C													
	C													

Figure 21.4 Calculating gross pay.

Operation Codes

The remainder of this chapter describes the various operation codes that can be written in RPG language. Simplified examples of common usage are also given. Refer back to Figure 21.2 for the summary of code specifications.

Because it is the intent of this text to present only the basic concepts of RPG, all possible uses of each operation will not be discussed. For complete details consult the reference manuals available from IBM.

Arithmetic Operations

Arithmetic operations are performed only on numerical fields or constants to produce a numerical result. Usually, two operands are involved: factor 1 and factor 2. A third field is produced as a result of operating on the two factors. The fields or constants may be used in any combination, thus:

1. Factor 1, factor 2, and the result may all be different fields with different names specified on the coding form. This type of operation is often used to crossfoot two fields in the same record, such as $A + B = C$.

2. Factor 1, factor 2, and the result may all be the same field. For example, a result obtained from a previous operation may be added to itself and the sum placed in the same field.

3. Factor 1 and factor 2 may be the same field but different from the result field. For example, a field may be multiplied by itself with the product placed in the result field.

4. Either factor 1 or factor 2 may be the same as the result field. Thus a previously obtained result can be combined with another field to produce the changed result in the same field, for example, to accumulate totals from a number of input records: *Field A + total = total.*

The length of any field involved in an arithmetic operation cannot exceed 15 digits, including the result field. In actual practice, the great majority of record fields are designed to contain values associated with commercial applications. Fields of as many as eight positions are rare, since a field of this size can contain quantities up to 99,999,999 or values up to $999,999.99. As previously described, the number of decimal positions in a field is specified in column 52 of the calculation form. Decimal alignment is automatic in RPG. Therefore, if addition of the two fields 123.456 and 0.00697 is specified, the correct sum of 123.46297 will result.

However, care must be taken to be sure the result field has the proper number of positions to contain the calculated answer. For example, if the factors 123.456 and 0.00697 are added and the result placed in a five-position field, the result will be 123.46 with the last three decimal positions dropped.

The dropping of most significant digits is more serious. This can occur when positions to the left overflow the capacity of the result field. For example, assume the factors 46952 and 53106 are added to obtain the sum of 100058. If a five-position result field is specified, the stored result will be 58 with the leftmost digit lost. It is not possible in RPG to detect such an overflow condition by indicators.

In the following descriptions, the RPG mnemonic code is given in parentheses after the name of the operation.

Add (ADD)

The contents of the field named as factor 2 are added algebraically to the contents of the field named as factor 1. The sum is stored in the field named as the result with the proper sign in the zone portion of the units position.

Factors 1 and 2 are not changed by the operation.

Zero and Add (Z-ADD)

The field named as the result is first set to zeros. Then, the contents of the field named as factor 2 are added to the result. Factor 1 is not used; factor 2 is unchanged.

In the following examples, the first line of each item shows the RPG coding. The result of the operation on various data appears with the corresponding line of code.

Examples:

	Factor 1	*Op. Code*	*Factor 2*	*Result*	*Field Length*	*Dec. Pos.*	*Half Adj.*
1.	FIELDA	ADD	FIELDB	SUM	7	3	H
	123.45+	+	0.1234+	123.573+			
	123.45+	+	0.1235+	123.574+			
2.	FIELDA	ADD	SUM	SUM	6	2	
	123.45+	+	123.573+	247.02+			
3.	SUM	ADD	SUM	SUM	6	1	
	123.4+	+	123.4+	246.8+			
4.		Z-ADD	HRS	TOTAL	4	1	
		+	38+	38.0+			
5.	FIELD1	ADD	FIELD1	RESULT	6	2	
	123.45+	+	123.45+	246.90+			
6.	FIELDX	ADD	FIELDY	FIELDZ	6	2	
	0.123+	+	123.45−	123.327−			
7.		Z-ADD	−123.4	RESULT	6	2	
		+	123.4−	123.40−			
8.	−123.4	ADD	ACTUAL	TOTAL	5	1	
	123.4−	+	1001.6−	1125.0			

Subtract (SUB)

The contents of the field named as factor 2 are algebraically subtracted from the contents of the field named as factor 1. The remainder is stored in the field named as the result with the proper sign placed in the zone portion of the units position. Factors 1 and 2 are not changed.

Zero and Subtract (Z-SUB)

The field named as the result is set to zero. Then, the contents of the field named as factor 2 are subtracted from the result. The effect is to store factor 2 in the result field and change the sign. Factor 1 is not used.

Examples:

	Factor 1	*Op. Code*	*Factor 2*	*Result*	*Field Length*	*Dec. Pos.*	*Half Adj.*
1.	FIELDA	SUB	FIELDB	REMDR	7	2	H
	123.45+	−	0.123+	123.33+			

	Factor 1	*Op. Code*	*Factor 2*	*Result*	*Field Length*	*Dec. Pos.*	*Half Adj.*
2.	FIELDA	SUB	REMDR	REMDR	6	2	
	123.45+	—	123.327+	0.12+			
3.	TOTAL	SUB	TOTAL	TOTAL	6	1	
	123.4+	—	123.4+	0.0+			
	123.4+	—	123.4—	246.8+			
	123.4—	—	123.4+	246.8—			
4.		Z-SUB	QUAN	TOTAL	4	0	
		—	123+	123—			
		—	123—	123+			
5.	FIELD1	SUB	FIELD1	RESLT	6	2	
	123.45+	—	123.45+	0.00+			
6.	FIELDX	SUB	FIELDY	FIELDZ	6	2	
	0.123—	—	123.45+	123.57—			

Multiply (MULT)

The contents of the field named as factor 1 are multiplied by the contents of the field named as factor 2. The product is placed in the field named as the result. The sign of the result is plus if both factors have like signs, and minus if they have unlike signs. Factors 1 and 2 are unchanged.

Examples:

	Factor 1	*Op. Code*	*Factor 2*	*Result*	*Field Length*	*Dec. Pos.*	*Half Adj.*
1.	FIELD1	MULT	FIELD2	PROD	5	0	
	123+	x	12+	1476+			
	986+	x	75—	73950—			
2.	FIELDA	MULT	FIELDB	FIELDC	6	2	H
	123.7+	x	0.18—	22.27—			
	123.7+	x	0.01+	1.24+			
	123.7—	x	0.001—	0.12—			

Divide (DIV)

The contents of the field named as factor 1 are divided by the contents of the field named as factor 2. The quotient is stored in the field named as the result. The sign of the quotient is plus if both factors have like signs, and minus if they have unlike signs.

If factor 1 is zero, the quotient will be zero. Factor 2 cannot be zero. If it is, a halt occurs at execution time.

The length of the data fields must adhere to specific rules or invalid results will be obtained. The following formulae apply:

$$L_1 + (D_2 - D_1 + D_R) \leqslant 15$$
$$L_2 - (D_2 - D_1 + D_R) \leqslant 15$$

where L_1 = length of the data field specified in factor 1
L_2 = length of the data field specified in factor 2
D_1 = number of digits to the right of the decimal point in factor 1
D_2 = number of digits to the right of the decimal point in factor 2
D_R = number of digits to the right of the decimal point in the result field

If half adjusting is specified, the length formula involving factor 1 must be satisfied as follows:

$$L_1 + (D_2 - D_1 + D_R) \leqslant 14$$

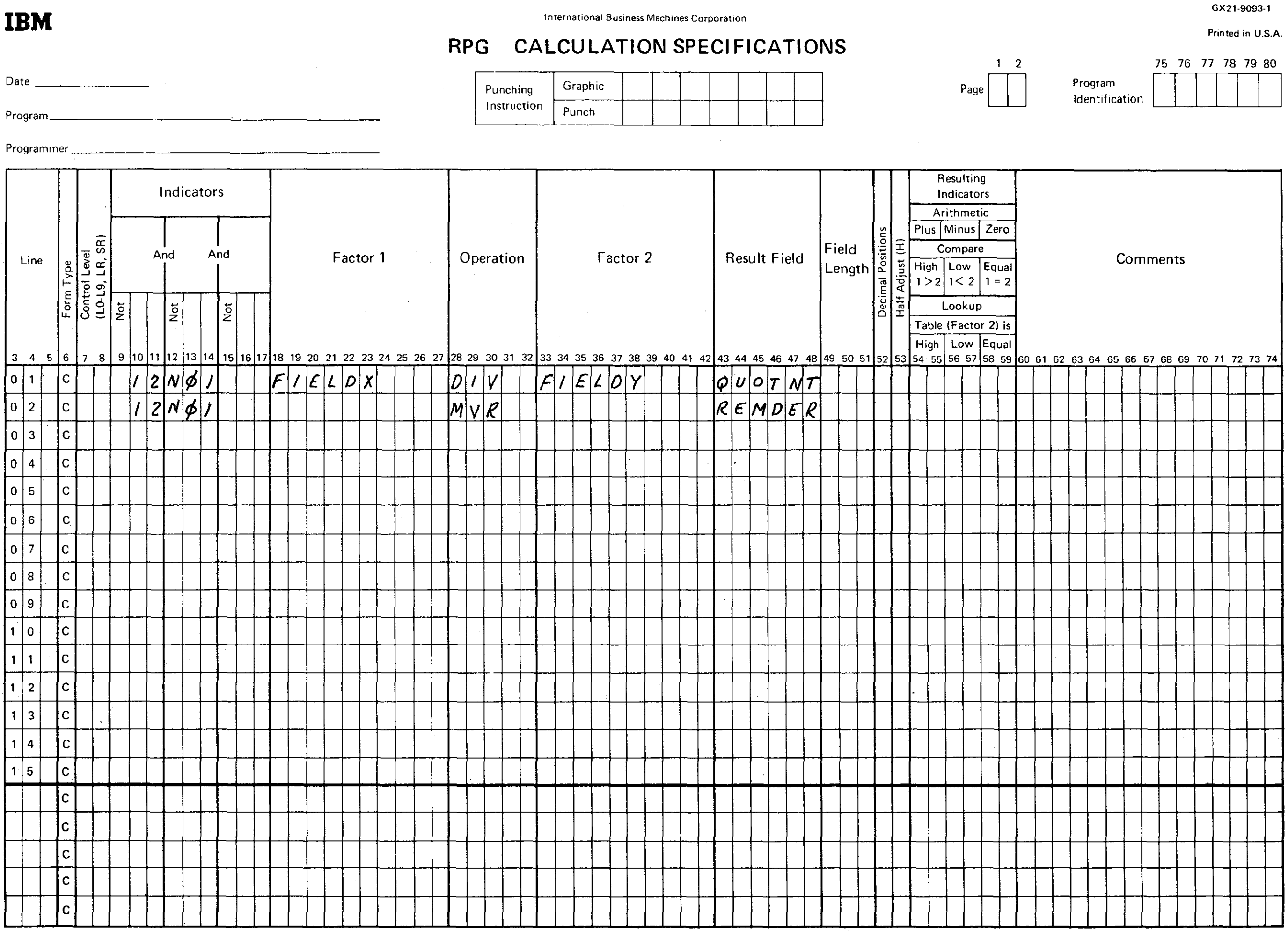
Examples:

	Op.			*Field*	*Dec.*	
Factor 1	*Code*	*Factor 2*	*Result*	*Length*	*Pos.*	*Rem.*
FIELDA	DIV	FIELDB	QUOT	6	0	
12345+	÷	123+	100+			45
12345+	÷	123–	100–			45
0+	÷	123+	0+			
123.45+	÷	123+	1.00+	6	2	45
123.45+	÷	1.23–	100.00–	6	2	45
0.869+	÷	1.4+	0.62+	4	2	1

Any remainder resulting from a divide operation is lost unless the move remainder operation is specified as the next operation. In this case the quotient cannot be half adjusted.

Move Remainder (MVR)

The remainder is moved to a separate field named as the result. Factors 1 and 2 must not be specified. The operation immediately follow a divide operation as shown in Figure 21.5.



IBM

International Business Machines Corporation

GX21-9093-1

Printed in U.S.A.

RPG CALCULATION SPECIFICATIONS

Date ______

Program ______

Programmer ______

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line	Form Type	Control Level (L0-L9, LR, SR)	Indicators		And		And		Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Arithmetic Plus / Compare High 1>2 / Lookup Table (Factor 2) is High	Minus / Low 1<2 / Low	Zero / Equal 1=2 / Equal	Comments
3 4 5	6	7 8	9 Not	10 11	12 Not	13 14	15 Not	16 17	18–27	28–32	33–42	43–48	49 50 51	52	53	54 55	56 57	58 59	60–74
0 1	C			12	N	Ø1			FIELDX	DIV	FIELDY	QUOTNT							
0 2	C			12	N	Ø1				MVR		REMDER							
0 3	C																		
0 4	C																		
0 5	C																		
0 6	C																		
0 7	C																		
0 8	C																		
0 9	C																		
1 0	C																		
1 1	C																		
1 2	C																		
1 3	C																		
1 4	C																		
1 5	C																		
	C																		
	C																		
	C																		
	C																		
	C																		

Figure 21.5 Move remainder.

The remainder is 15 positions in length. The number of decimal positions equals the number of positions in factor 2 of the divide operation. If the remainder is less than 15 positions, the leftmost zeros may be dropped. Resulting indicators may also be used for MVR.

Crossfoot (XFOOT)

Elements of an array are added together. The sum is placed in the field named as the result. Factor 2 contains the name of the array. Factor 1 is not used.

Move Operations

Move operations place all or a part of factor 2 in the result field. Factor 2 is not changed. Factor 1 is not used and must be left blank on the specification sheet. Resulting indicators are not used.

Move (MOVE)

This operation causes characters or digits from the field named as factor 2 to be placed in the field named as the result. Moving is done, character by character, beginning with the rightmost position of factor 2 and proceeding from right to left until all positions are placed in the result.

If factor 2 is longer than the result, excess positions in factor 2 are not moved. If the result field is longer than factor 2, excess positions to the left of the result field are not changed.

An alphameric field or constant is changed to a numerical field by placing it in a numerical result (0 in column 52). The digit portion of each character is converted to its corresponding numerical character and then moved to the result field. Blanks are moved as zeros. The zone portion of the rightmost character becomes the sign of the result.

Conversely, a numerical field or constant may be changed to an alphameric field by moving it to an alphameric result (blank in column 52). Examples of the MOVE operation are shown in Figure 21.6.

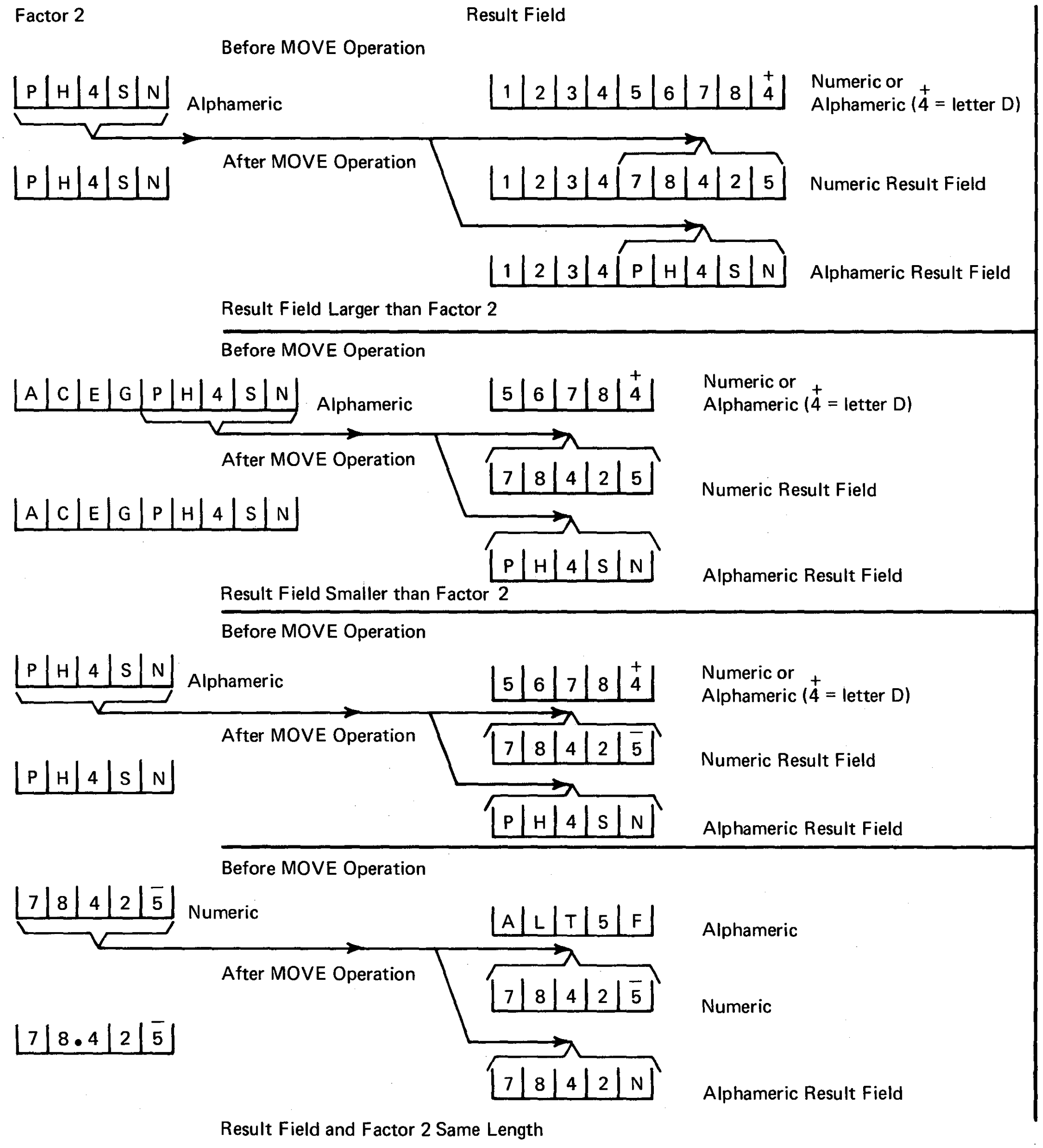


Figure 21.6 Move operations.

Move Left (MOVEL)

This operation is the same as the MOVE operation, except that factor 2 is left justified in the result field. If factor 2 is longer

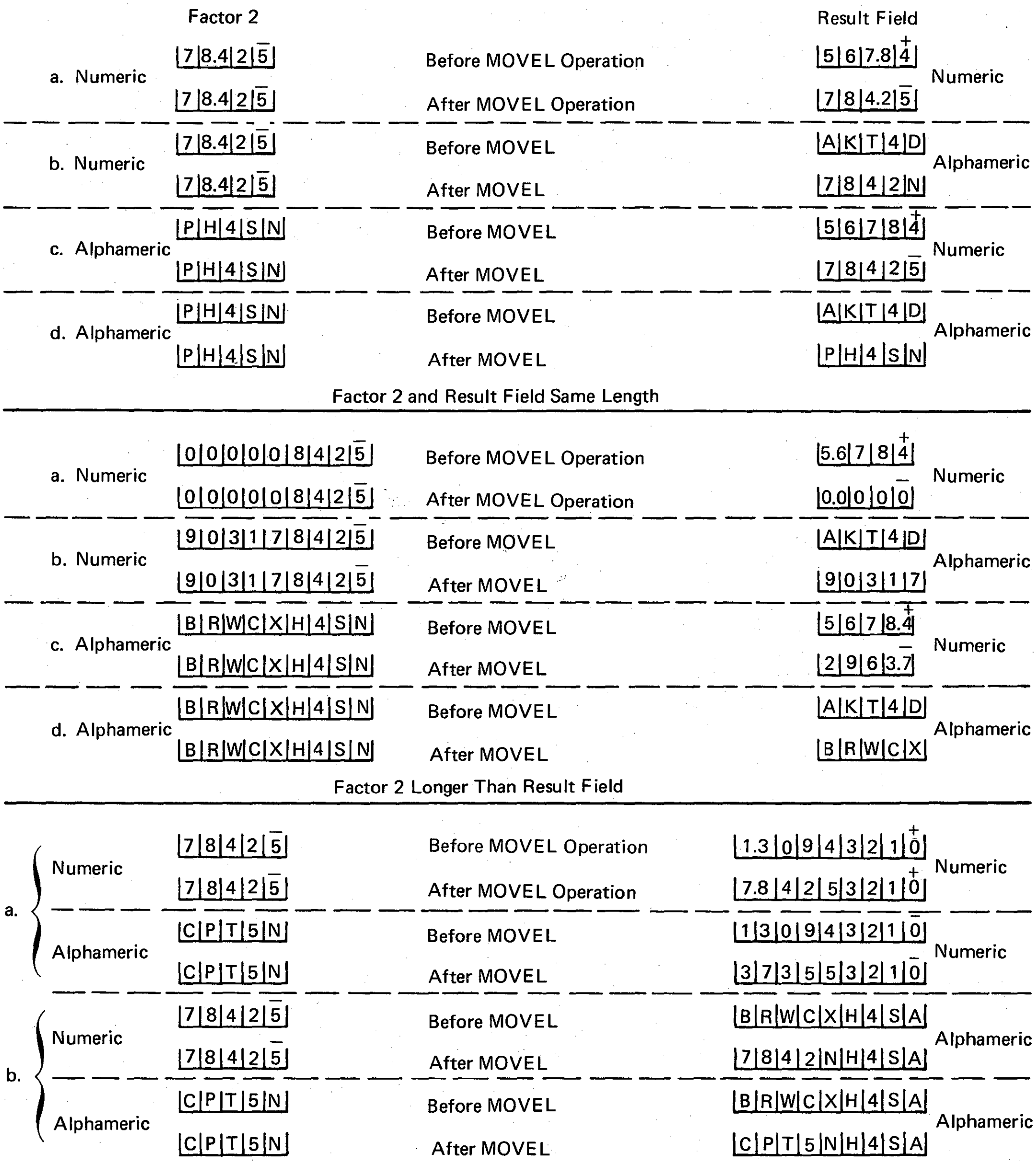


Figure 21.7 Move left.

than the result, the excess rightmost positions of factor 2 are not moved. If the result is longer, excess positions to the right of the result are not affected.

Alphameric and numerical fields are changed in the same manner as in the MOVE operation. Examples are shown in Figure 21.7.

Move Zone Operations

These operations affect either the zone or numerical portions of a character. They provide detailed flexibility to operate within the character structure, for example, to change the sign of a numerical field.

Four operations are available in RPG II:

Move High to High Zone (MHHZO)
Move High to Low Zone (MHLZO)
Move Low to Low Zone (MLLZO)
Move Low to High Zone (MLHZO)

In these operations, the word *high* refers to an alphameric field; the word *low* to a numerical field.

Figure 21.8 is a tabulation of the functions of these four operations.

MHHZO	Factor 2	Alpha	Bits 0 - 3 of leftmost byte of factor 2 are moved to bits 0 - 3 of leftmost byte of result.
	Result	Alpha	
MHLZO	Factor 2	Alpha	Bits 0 - 3 of leftmost byte of factor 2 are moved to bits 4 - 7 of rightmost byte of result.
	Result	Numeric	
	Factor 2	Alpha	Bits 0 - 3 of leftmost byte of factor 2 are moved to bits 0 - 3 of rightmost byte of result.
	Result	Alpha	
MLLZO	Factor 2	Alpha	Bits 0 - 3 of rightmost byte of factor 2 are moved to bits 0 - 3 of rightmost byte of result.
	Result	Alpha	
	Factor 2	Alpha	Bits 0 - 3 of rightmost byte of factor 2 are moved to bits 4 - 7 of rightmost byte of result.
	Result	Numeric	
	Factor 2	Numeric	Bits 4 - 7 of rightmost byte of factor 2 are moved to bits 0 - 3 of rightmost byte of result.
	Result	Alpha	
	Factor 2	Numeric	Bits 4 - 7 of rightmost byte of factor 2 are moved to bits 4 - 7 of rightmost byte of result.
	Result	Numeric	
MLHZO	Factor 2	Alpha	Bits 0 - 3 of rightmost byte of factor 2 are moved to bits 0 - 3 of leftmost byte of result.
	Result	Alpha	
	Factor 2	Numeric	Bits 4 - 7 of rightmost byte of factor 2 are moved to bits 0 - 3 of leftmost byte of result.
	Result	Alpha	

Figure 21.8 Move zone operations.

Compare and Testing Operations

By performing compare and testing operations, the computer can be programmed to follow one of several possible paths through a procedure. The selection of the proper path is made by comparing one item of stored data with another. Single characters, selected fields, or entire records can be compared. One of three possible conditions invariably results: a given field is always higher, lower, or equal to another.

The condition resulting from the comparison is registered by designated indicators. When the indicator is on, the machine can be instructed to *branch* to a high, low, or equal program routine. Such branch instructions are termed *conditional* branches and have the effect of telling the machine to GO TO another instruction *provided* the designated indicator is on. If the indicator is off, continue with the next instruction in the sequence.

As previously explained, the determination of whether one field or character is higher, lower, or equal to another is made according to the established collating sequence of the system. In RPG II, that sequence can be varied by using a special coding form.

Compare (COMP)

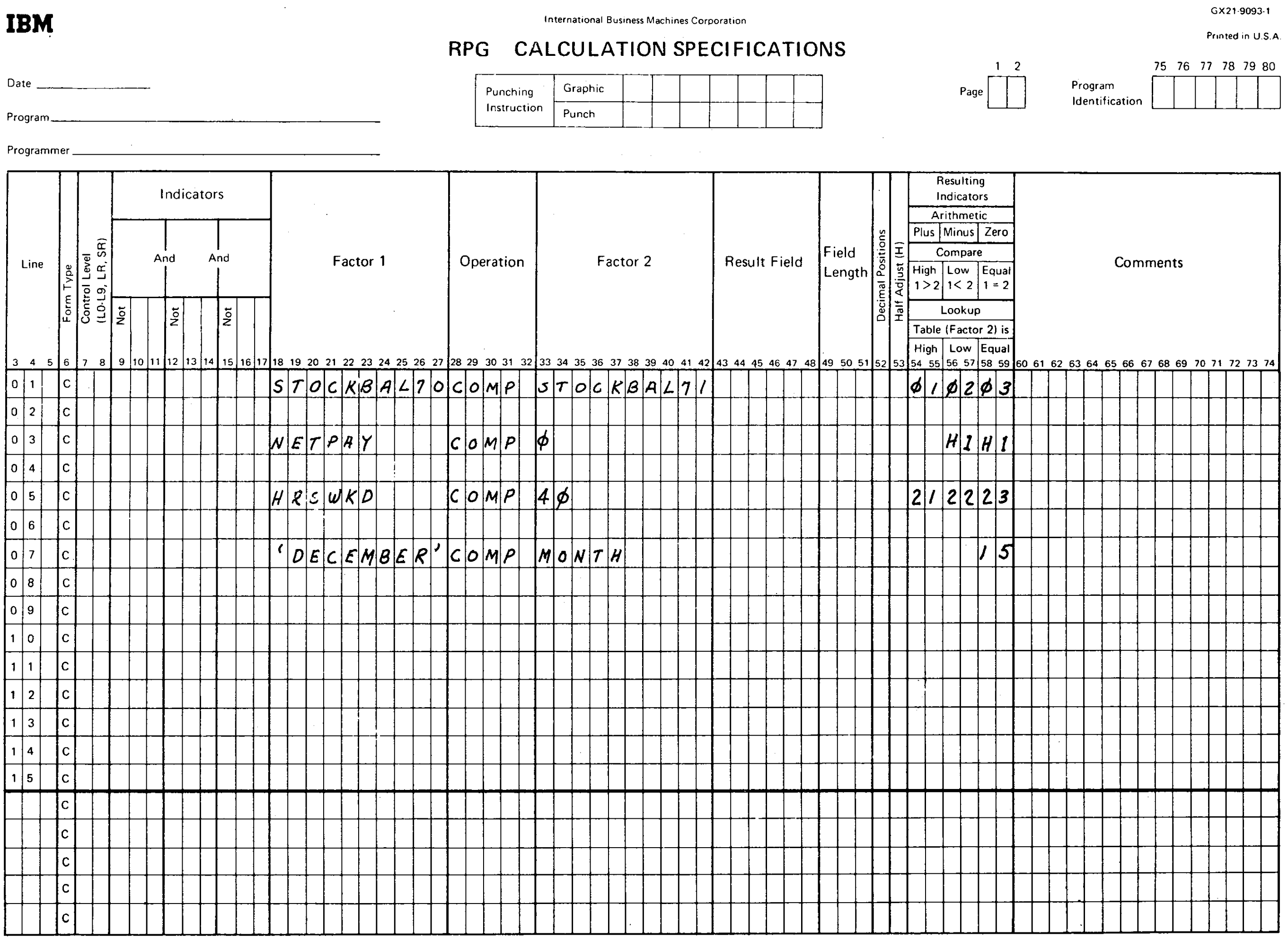
Factor 1 is compared with factor 2. As a result of the comparison, indicators are turned on as follows:

High:	Factor 1 is greater than factor 2
Low:	Factor 1 is less than factor 2
Equal:	Factor 1 equals factor 2

The record fields named as factors 1 and 2 are automatically aligned before they are compared. If the fields are alphameric, they are left-justified. If one field is shorter, the unused right positions are filled with blanks. Fields of unequal length to be compared may contain as many as 40 characters. Fields of equal length may contain a maximum of 256 characters.

Numerical fields to be compared are aligned by the decimal point. Any missing digits are filled in with zeros. Maximum field length is 15 digits.

Figure 21.9 shows a number of compare operations with named fields and constants. The item numbers below correspond with line numbers in the illustration.



IBM

International Business Machines Corporation

GX21-9093-1

Printed in U.S.A.

RPG CALCULATION SPECIFICATIONS

Date

Program

Programmer

Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR) (7-8)	Indicators: Not (9)	And (10-11)	Not (12)	And (13-14)	Not (15)	(16-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Arithmetic Plus / Compare High 1>2 / Lookup Table (Factor 2) is High (54-55)	Minus / Low 1<2 / Low (56-57)	Zero / Equal 1=2 / Equal (58-59)	Comments (60-74)
01	C								STOCKBAL70	COMP	STOCKBAL71					Ø1	Ø2	Ø3	
02	C																		
03	C								NETPAY	COMP	Ø						H1	H1	
04	C																		
05	C								HRSWKD	COMP	4Ø					21	22	23	
06	C																		
07	C								'DECEMBER'	COMP	MONTH							15	
08	C																		
09	C																		
10	C																		
11	C																		
12	C																		
13	C																		
14	C																		
15	C																		
	C																		
	C																		
	C																		
	C																		
	C																		

Figure 21.9 Compare operations.

Line 01. The quantity of an item in stock at the end of 1970 is compared with the quantity at the end of 1971. If the 1970 balance is higher, indicator 01 is turned on; if the quantity is lower, indicator 02 is turned on; if the quantities are equal, indicator 03 is turned on. Following operations are conditioned upon the condition of the indicators. The logical paths followed can be flowcharted as follows:

Line 03. This instruction might be the last in a routine to calculate employee net pay. After calculation, net pay is compared with the constant zero. If an employee's pay is lower than or equal to zero, an error has been made and halt indicator H1 is turned on. Program execution is halted for corrective action.

Line 05. The hours worked field is compared with the numerical constant 40. If an employee works more than 40 hours, indicator 21 is turned on to select a routine to calculate overtime. Separate routines can be selected when hours worked is equal to or less than 40. The same results can be obtained by subtracting the numerical constant 40 from hours worked.

Line 07. The alphameric constant 'DECEMBER' is compared with the month field. If an equal comparison occurs, a special routine is selected by turning on indicator 15. In this case only one of two possible routines can be followed.

Test Zone (TESTZ)

This operation is used to test the zoning of the leftmost character in the result field. The field must be alphameric. The zone portions of the characters &, A-I cause the plus indicator to turn on. The zone portions of the characters —, J-R causes the minus indicator to turn on. When all other characters are tested, the zero indicator is turned on. Factors 1 and 2 are not used in this operation.

Setting Indicators

The preceding sections have shown that indicators can be turned on by the results of calculation, comparison, or zone testing. Indicators can also be set by special operations. These operations enable the programmer to preset indicators in anticipation of predetermined conditions. For example, at the start of a program, indicators are normally turned off to prevent incorrect branching to exception procedures.

Set On (SETON)

Any indicators in column 54 through 59 are turned on.

Set Off (SETOF)

Any indicators in columns 54 through 59 are turned off.

Branching Operations

Operations are normally performed in the sequence in which they are written on the specification forms. However, there are many instances in normal procedures when it is desirable to execute instructions in other than the given order. For example, for repetitive functions, the same program routine can be repeated a number of times. Or, for exception routines, certain specified instructions may be skipped and not executed at all.

Two elements are involved in a program branch. A special operation, GOTO, indicates that the next operation in sequence is not to be executed. Instead the machine is to GOTO another operation identified in factor 2. The program execution is then continued from that point. The destination operation is given as TAG, with the destination name in factor 1.

Go To (GOTO)

This operation enables the programmer to specify branching operations from one point in a program to another. The branch may be to an operation line before or after the GOTO line. However, a branch cannot be made from a calculation that is not conditioned by a control level indicator to one that is, or vice versa.

Factor 2 contains the name of the TAG operation code line to where the branch is to be made. Factor 1 and the result field are not used.

The GOTO operation may be conditioned by any indicators except 1P. The branch is always made if the operation is not conditioned.

Tag (TAG)

The TAG operation code line names the destination of the GOTO operation. Factor 1 contains the destination name. The same name cannot be used for more than one TAG operation, since a branch to more than one point in a program is impossible. The TAG name cannot be used to name any field used in the program. Factor 2 and the result field are not used.

The TAG operation may not be conditioned by indicators in columns 9 through 17, except control level indicators. These indicators must be used when branching is to occur only when information in a control field has changed.

Figure 21.10 shows an example of GOTO and TAG coding to execute exception routines. A flowchart of the procedure logic is shown in Figure 21.11.

Line 01. Field B is subtracted from field A. If the remainder is minus, indicator 10 is turned on.

Line 02. Branch to routine 1 if indicator 10 is on. If it is off, continue to the next operation.

Line 03. Multiply field B by the numerical constant 4.

Line 04. Branch to routine 1, beginning on code line 09. This branch is unconditional.

Lines 09-12. Continue with routine 1.

Line 13. Branch to routine 2 if indicator 15 is off. If indicator 15 is on, continue to line 14.

Line 14. Test field C for zone in the leftmost character. If the result is plus or zero, turn on indicator 20.

Line 15. If indicator 20 is on, go to the END routine; if off, continue to line 16 and END.

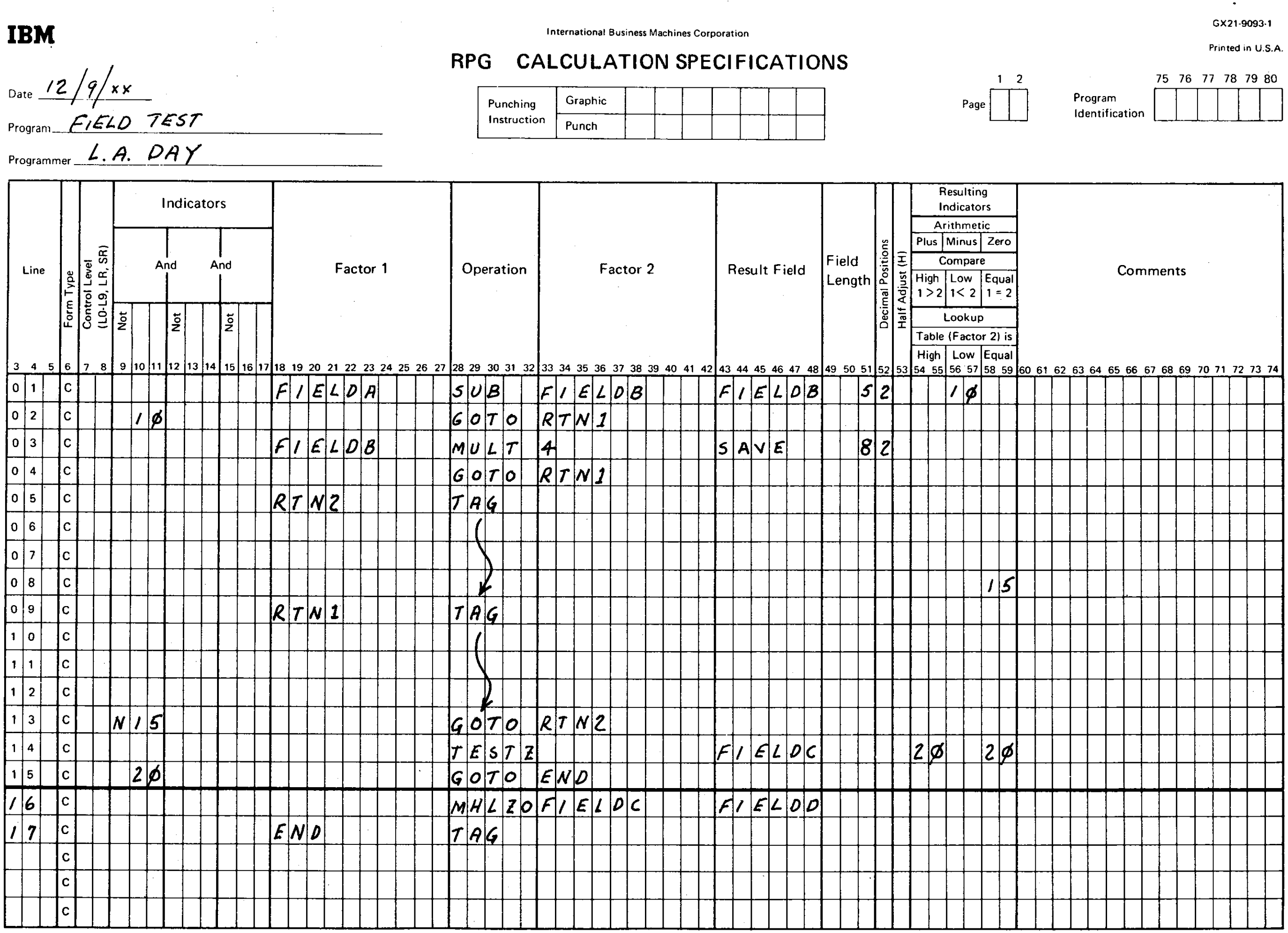
The example in Figure 21.12 shows how branching operations can be used to eliminate coding repetitive operations. The coding will produce a program "loop" that will be repeated a specified number of times. To control the number of repeats, a field is set aside as a counter. Each time the computer follows the loop of instructions, a 1 is added to the counter. The amount in the counter is then compared against a predetermined quantity. When the count equals the given quantity, the loop has been executed the specified number of times. A branch is then effected out of the loop to continue the program execution from that point on.

Line 01. The program loop is entered here. The TAG operation names the beginning at DOOVER.

Line 02. Field A is added to itself.

Line 03. 1 is added to the count field.

Line 04. Count is compared to 5. If count equals 5, indicator 20 is turned on.



IBM

International Business Machines Corporation

GX21-9093-1

Printed in U.S.A.

RPG CALCULATION SPECIFICATIONS

Date 12/9/xx

Program FIELD TEST

Programmer L.A. DAY

Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR) (7-8)	Indicators (9-11)	And (12-14)	And (15-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Arithmetic Plus / Compare High 1>2 / Lookup Table (Factor 2) is High (54-55)	Minus / Low 1<2 / Low (56-57)	Zero / Equal 1=2 / Equal (58-59)	Comments (60-74)
01	C					FIELDA	SUB	FIELDB	FIELDB	5	2			1Ø		
02	C		1Ø				GOTO	RTN1								
03	C					FIELDB	MULT	4	SAVE	8	2					
04	C						GOTO	RTN1								
05	C					RTN2	TAG									
06	C															
07	C															
08	C														15	
09	C					RTN1	TAG									
10	C															
11	C															
12	C															
13	C		N15				GOTO	RTN2								
14	C						TESTZ		FIELDC				2Ø		2Ø	
15	C		2Ø				GOTO	END								
16	C						MHLZO	FIELDC	FIELDD							
17	C					END	TAG									
	C															
	C															
	C															

Figure 21.10 GO TO and TAG coding.

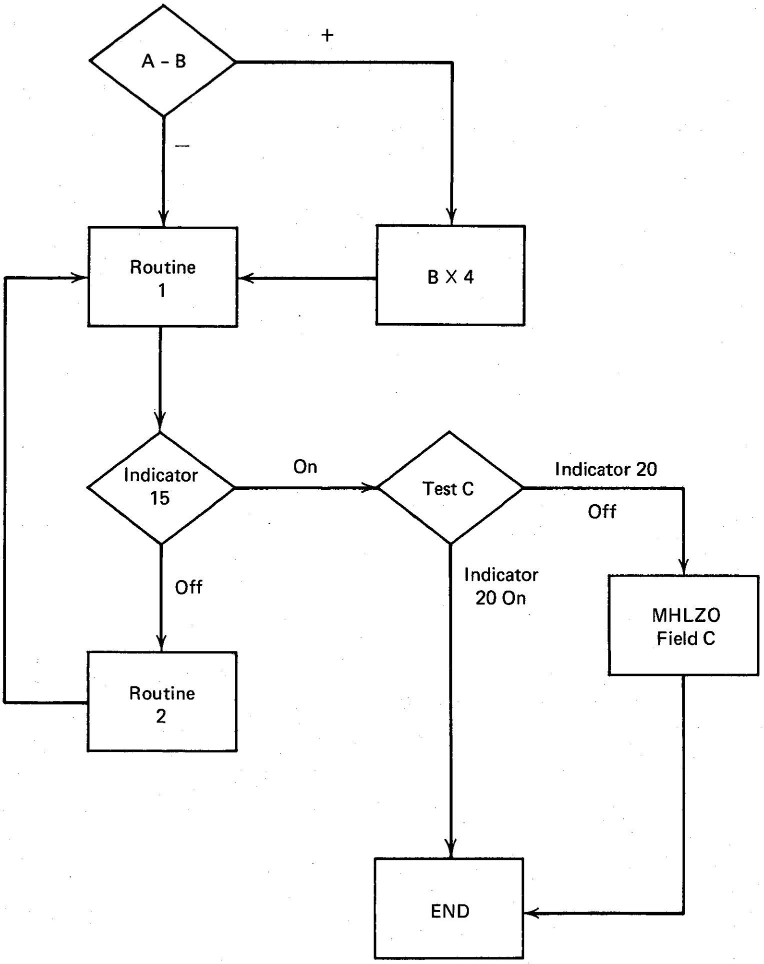


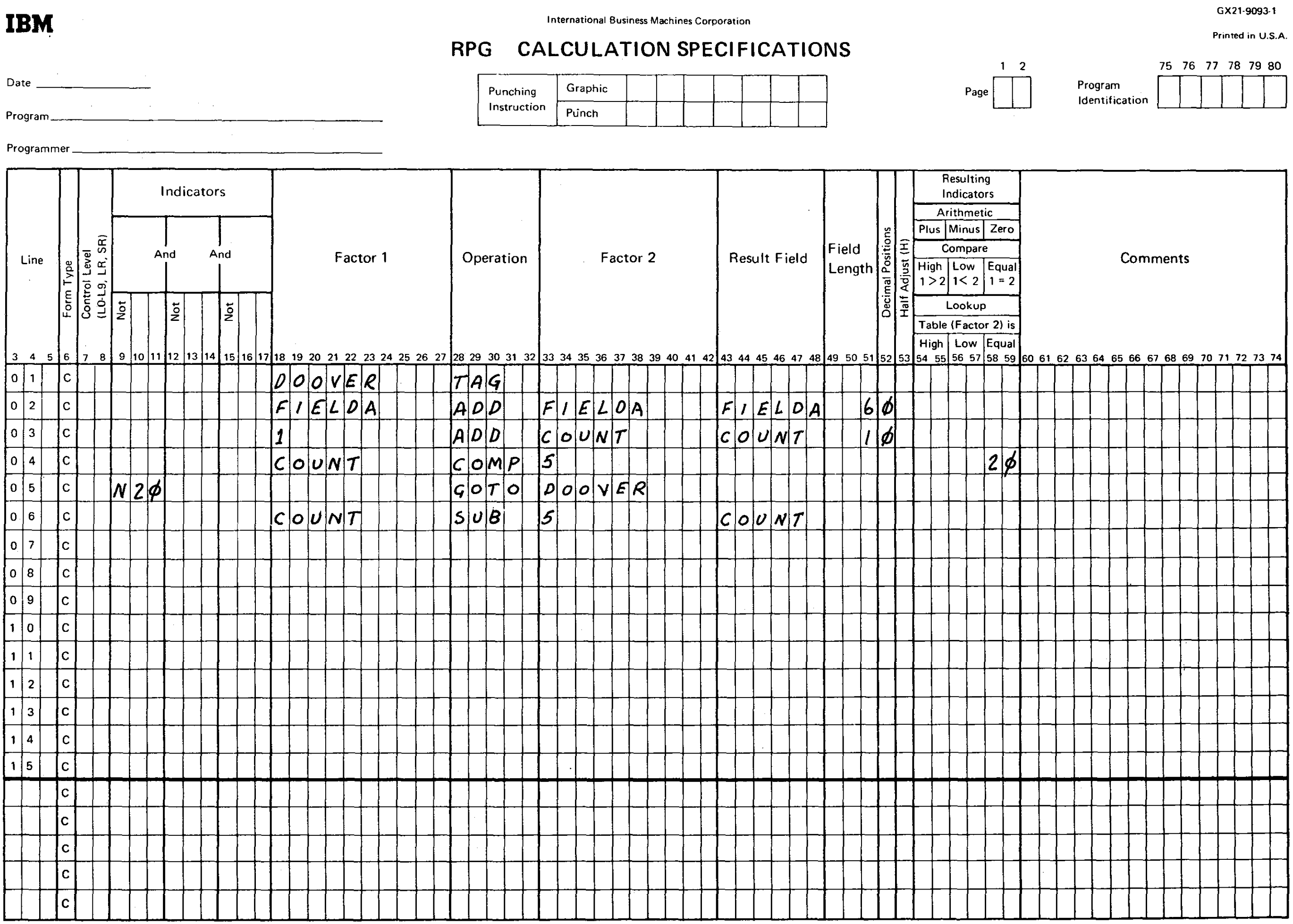
Figure 21.11 GO TO and TAG flowchart.

Line 05. If indicator 20 is off, go back to DOOVER. If indicator 20 is on, the loop has been executed 5 times.

Line 06. Subtract 5 from count to set the field back to zero in preparation for the next execution of the loop.

The routine has the effect of multiplying Field A by 5.

Subroutine Operations Subroutines are written in the same manner as other calculation specifications in the RPG source program. The subroutines are then compiled with the source program and become part of the



IBM

International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

GX21-9093-1

Printed in U.S.A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page (1 2)

Program Identification (75 76 77 78 79 80)

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Plus / High 1>2 (54-55)	Minus / Low 1<2 (56-57)	Zero / Equal 1=2 (58-59)	Comments (60-74)
01	C			DOOVER	TAG									
02	C			FIELDA	ADD	FIELDA	FIELDA	6	Ø					
03	C			1	ADD	COUNT	COUNT	1	Ø					
04	C			COUNT	COMP	5							2Ø	
05	C		N2Ø		GOTO	DOOVER								
06	C			COUNT	SUB	5	COUNT							
07	C													
08	C													
09	C													
10	C													
11	C													
12	C													
13	C													
14	C													
15	C													
	C													
	C													
	C													
	C													
	C													

Figure 21.12 Branching operations.

resulting object program. One subroutine can be compiled with any number of different source programs by including the cards punched for the subroutine as part of the various source decks.

Each subroutine must be given an identifying name and must begin with the operation BEGSR and end with the operation ENDSR. These two operation codes mark the limits of the subroutine and separate it from the other elements of the calculation specifications.

All subroutines must be written in specification lines following all operations conditioned by control level indicators specified in columns 7 and 8. Subroutine lines are always identified by the letters SR in columns 7 and 8.

Execute Subroutine (EXSR)

This operation causes the designated subroutine to be executed. EXSR may be written anywhere in the source program. After the subroutine has been executed, the operation in the line following the EXSR operation is executed.

The EXSR may be conditioned by any indicators. Then, the subroutine can only be executed if all conditions are satisfied. Factor 2 contains the name of the subroutine to be executed. This same name must appear on a BEGSR line.

Begin Subroutine (BEGSR)

This operation serves as the beginning point of the subroutine. Factor 1 must contain the name.

End Subroutine (ENDSR)

This operation must be the last statement of the subroutine. It defines the end of the routine. The ENDSR automatically causes a branch back to the next operation line after the EXSR line.

Example: Use of Subroutines

Figure 21.13 is a flowchart showing logic of subroutine use in calculating net pay in a payroll application. Three aspects of the calculation procedure are shown.

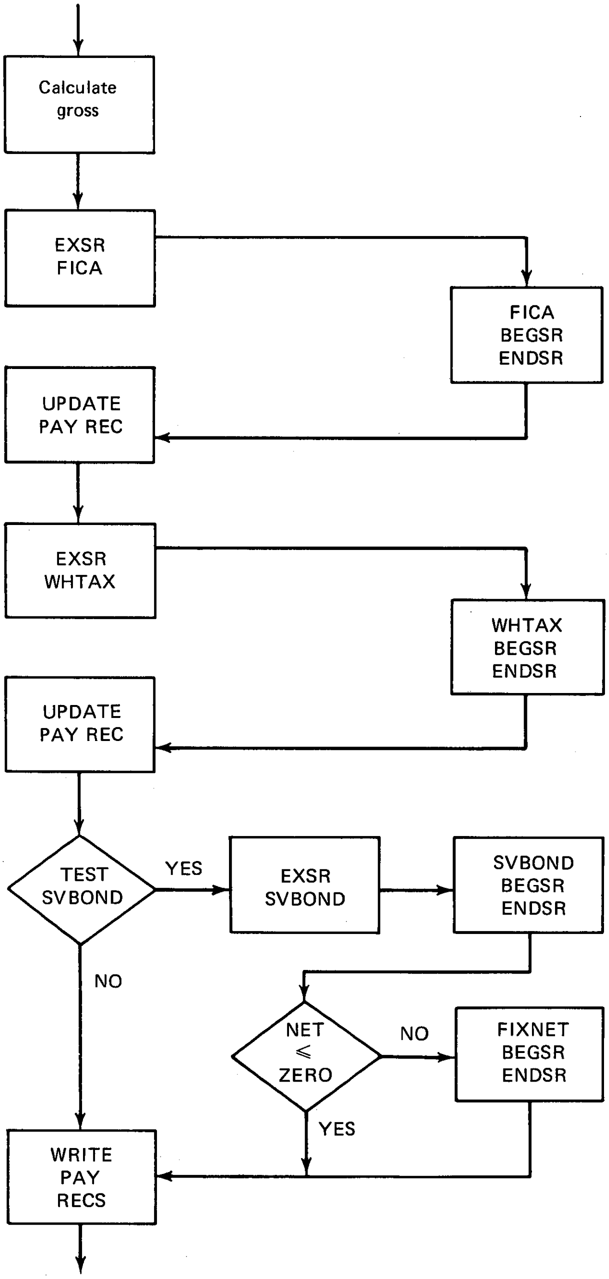


Figure 21.13 Subroutine logic flowchart.

1. *F.I.C.A.* After gross pay amount is calculated, a branch is effected to the F.I.C.A. subroutine. The branch is unconditional.

The subroutine can include all instructions to calculate the social security tax, including the year-to-date accumulation and the test for limit of payment according to current federal tax regulations.

When F.I.C.A. is calculated, the machine is directed to the next instruction in the main program. The amounts obtained by the subroutine are placed in the employee pay record. If no tax is calculated, the current pay period tax field is filled with zeros. The year-to-date tax field is unaffected.

2. *Withholding Tax.* An EXSR operation causes a branch to the WHTAX subroutine where federal income tax for the employee is calculated. The branch is unconditional. When the subroutine is completed, the main program is entered at the next instruction. The pay record is again updated to include the result of the tax calculation.

3. *Savings Bond.* It is assumed that in this application, employees may have authorized amounts deducted from their pay toward the purchase of a U.S. savings bond. The amount of each authorized deduction is included in the master records for the employee.

If the employee has authorized a deduction, an indicator is on, and the branch is effected to the SVBOND subroutine. The branch is conditional upon the designated indicator. The subroutine should contain all instructions needed to do the accounting for bond deductions.

The following formula is calculated at the end of the subroutine:

GROSS PAY − (FICA + WHTAX + SAVINGS BOND) = NET PAY

The net pay field is then tested to determine if the amount is less than or equal to zero. Net pay could be a minus or zero quantity if an employee had not worked long enough in the current period to earn the deduction amount. In this case, no paycheck can be written and a corresponding error would appear on the pay register.

If net pay is less than or equal to zero, a FIXNET routine is entered to add the bond deduction back to the employee's pay and to reverse the bond accounting entries. At the end of the routine, the main program is reentered to prepare the output documents for printing and punching.

The use of subroutines in the above procedure provides a distinct advantage of making the program modular. That is, certain parts of the program can be changed or eliminated without affecting the main program. If federal tax laws are modified, the tax subroutines can be readily altered without affecting other parts of the application. Also, should bond deductions be discontinued, the routine can be eliminated by using one of the external indicators available in the RPG II system. Branching can be made conditional on the specified indicator.

Note also that the fix routine corrects a potential error automatically. The routine could also include provision for printing out a list of those employees who receive no pay for a period or who are over-deducted.

Other Operations

The RPB II language includes a number of additional operation codes that will not be completely explained in this text. For the most part, these are supplementary features not always required for programming an application.

Look-up (LOKUP).

The operation executes a search for an item in a table. Factor 1 is the search word; factor 2 is the table name.

Exception (EXCPT).

The operation allows records to be written out at the time calculations are being performed. A number of multiple records can be produced from a single output record, for example, a given quantity of duplicate shipping labels for one customer.

Force (FORCE).

A record can be read from any file thereby overriding the standard sequence of input operations.

Debug (DEBUG).

Placed at any or several points in the program, this operation causes a printout of all indicators that are on at that point in the program. The contents of a named field can also be printed. The operation is an aid in tracing programming errors during the testing of the object program from the RPG assembly.

Questions and Exercises

1. List the four parameters of all calculations in RPG II in the order in which they are placed on one line of the specification sheet.
2. How are RPG II indicators similar to pilot selectors and coselectors in an accounting machine?
3. What is a constant? How is it used?
4. What is an operation code? List some RPG II examples.
5. What are result indicators?
6. Given the data supplied, what result is obtained from the following operations.

Field A	*Oper.*	*Field B*	*Field Length*	*Dec. Pos.*	*Half Adj.*
16.24+	ADD	0.56923+	5	3	
	ZADD	123	3	0	
12.34–	ADD	1532.00	6	3	
6283+	SUB	7431.0+	5	1	
12.782	SUB	14.632+	5	3	
246–	MULT	246+	6	1	H
2468–	MULT	0.55	5	1	H
2468+	DIV	24–	3	0	
463.82	DIV	1.234	4	2	

7. Prepare the calculation specification sheet to continue the case study problem following chapter 20.

RPG II OUTPUT-FORMAT SPECIFICATIONS AND EXAMPLES 22

Output-format specifications describe all output files. Entries are of two types:

1. File identification entries describe the files and the records to be produced within the files.

2. Data description entries locate the data within the records.

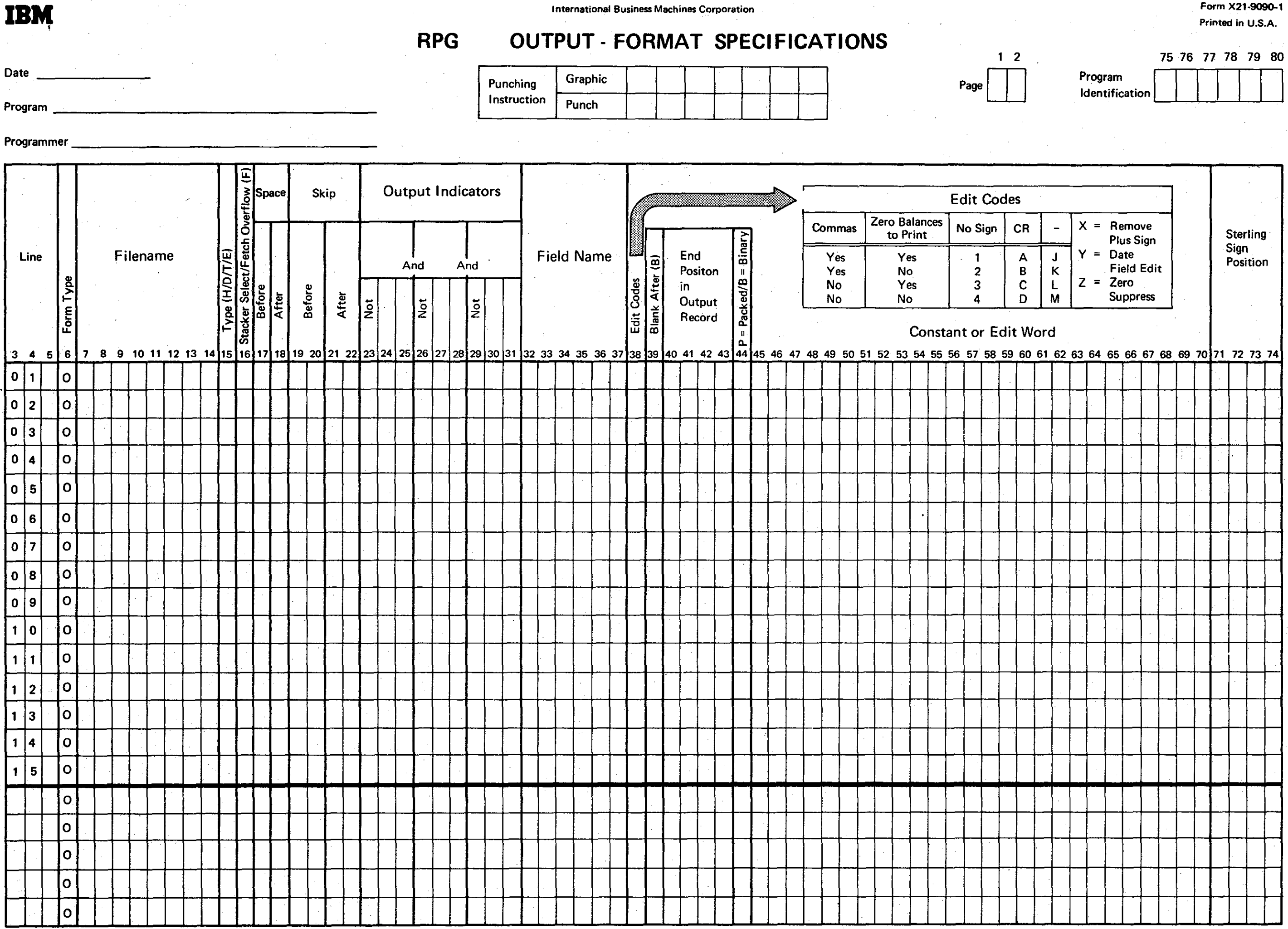
Output-Format Specifications

Like other RPG forms, the Output-Format Specification sheet is divided into 80 columns as shown in Figure 22.1. This section describes the entries which can be made on the form.

Columns 1 through 7 and 75 through 80. Entries for these columns are the same as described in chapter 21.

Columns 7 through 14. This entry identifies the output file by name. Construction of the name must adhere to the same rules as those established for the file description form. The name must be written on the first line that identifies the file.

Column 15. This entry identifies the type of record to be written: H for header, D for detail, and T for total. The record may be either a printed line or a 96-column punched card. An E entry indicates a record to be printed or punched during calculation time.



IBM

International Business Machines Corporation

Form X21-9090-1
Printed in U.S.A.

RPG OUTPUT - FORMAT SPECIFICATIONS

Date ______________

Program ______________

Programmer ______________

Punching Instruction	Graphic							
	Punch							

Page 1 2 □□

Program Identification 75 76 77 78 79 80 □□□□□□

Line (3 4 5) | Form Type (6) | Filename (7–14) | Type (H/D/T/E) (15) | Stacker Select/Fetch Overflow (F) (16) | Space: Before (17), After (18) | Skip: Before (19 20), After (21 22) | Output Indicators: Not (23), (24 25); And Not (26), (27 28); And Not (29), (30 31) | Field Name (32–37) | Edit Codes (38) | Blank After (B) (39) | End Positon in Output Record (40 41 42 43) | P = Packed/B = Binary (44) | Constant or Edit Word (45–70) | Sterling Sign Position (71 72 73 74)

Edit Codes

Commas	Zero Balances to Print	No Sign	CR	-	
Yes	Yes	1	A	J	X = Remove Plus Sign
Yes	No	2	B	K	Y = Date Field Edit
No	Yes	3	C	L	Z = Zero Suppress
No	No	4	D	M	

Line			Form Type
0	1		O
0	2		O
0	3		O
0	4		O
0	5		O
0	6		O
0	7		O
0	8		O
0	9		O
1	0		O
1	1		O
1	2		O
1	3		O
1	4		O
1	5		O
			O
			O
			O
			O
			O

Figure 22.1 Output-format specifications.

Record type must be entered in the order in which the records are to be produced: heading, detail, total, and EXCPT. E may not be specified for a combined file.

Column 16. An entry in this column is used:

1. to select the stacker for specified cards;

2. to indicate that the page overflow routine can be used at this point.

Entry	*Explanation*
Blank	Primary cards automatically fall in stacker 1, secondary cards in stacker 4.
1-4	Indicates selected stacker.
F	Fetch overflow.

The F entry in the column causes the overflow routines to be performed in advance of their normal execution. In some instances, it may be desirable to print total and detail lines at the top of a new page rather than after the heading lines conditioned by overflow have been printed.

Columns 17 and 18. Entries in these columns indicate the number of line spaces to be taken before or after printing. No spacing, single-spacing, double-spacing, or triple-spacing is indicated by 0, 1, 2, or 3 respectively. An entry in column 17 indicates spacing before the line is printed; entry in column 18 indicates spacing after printing.

Columns 19 through 22. Skipping is used when more than three lines of space are needed between printed lines. Spacing or skipping should normally occur after printing to avoid delay waiting for the form to be positioned. Entries in these columns indicate the line number for the form to which the skip is to be made.

On the System/3, overflow skipping to the next page takes place automatically after line 60 is printed. Therefore, when using standard 66-line paper, six-line margins are placed at the top and bottom of each page. Procedures that need different form specifications must be described with Line Counter Specifications, a special form for this purpose.

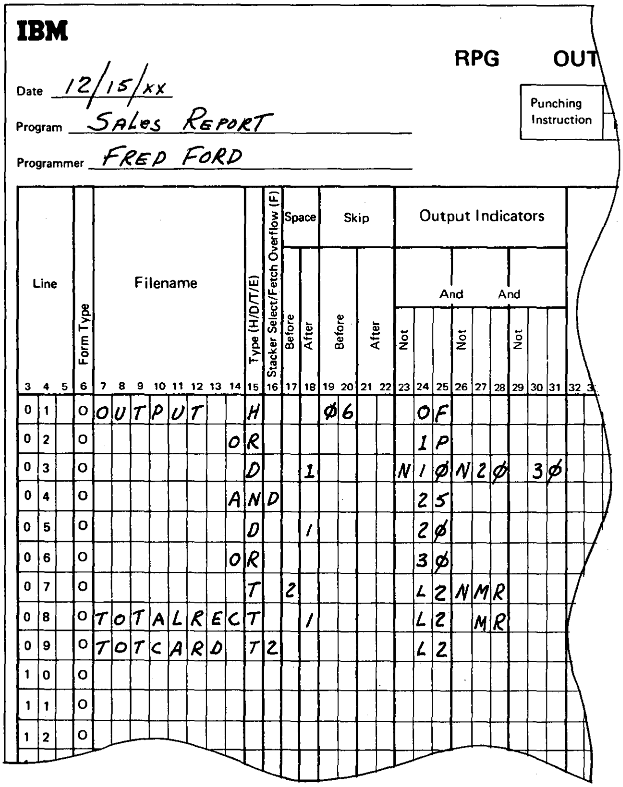
Entry	*Explanation*
0-99	Lines 0-99
A0-A9	Lines 100-109
B0-B2	Lines 110-112

Entries in columns 19 and 20 indicate skipping before a line is printed; entries in columns 21 and 22 indicate skipping after the line is printed.

Columns 23 through 31. Output indicators are entered in these columns to specify the conditions under which a record is to be punched or printed. As many as three indicators can be specified. The indicators are written on the same line as the name of the record to be produced as output.

Entry	*Explanation*
01-99	Any resulting indicators, field indicators, or record identifying indicators previously identified. Thus a record may be punched or printed when the result of a calculation is plus, minus, or zero; when the result of a comparison is high, low, or equal; and so on.
L1-L9	Any control level indicators previously specified.
H1-H9	Halt indicators. When a condition occurs that stops the program, a record can be produced as a result of the halt; for example, a message to the operator.
U1-U8	An external indicator set prior to program execution. For example, different report formats can be produced in separate runs from the same program and input data. External indicators condition printing at the option of the operator or programmer.
OA-OG, OV	Any overflow indicator previously assigned to this file. Totals may be printed at the end of a page, for example, even though no control breaks have occurred in the input files.
MR	Matching record indicator. Used to print or punch a record when records match as previously specified.
LR	Last record indicator. Can be used to print or punch the total at the end of a job.
1P	First page indicator. Can be used to print headings or other identifying information on the first page of a report.

As many indicators as may be needed can be specified by using more than one line of the form. AND, written in columns 14 through 16, establishes the relationship between the two lines of indicators. OR relationships can also be specified by writing OR in columns 14 and 15 of a subsequent line and entering the alternate indicators on that line. Figure 22.2 is an example of using indicators to condition output records. Item numbers below refer to the corresponding lines of the specification form.



IBM

RPG OUT

Punching Instruction

Date 12/15/xx

Program SALES REPORT

Programmer FRED FORD

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T/E) (15)	Stacker Select/Fetch Overflow (F) (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23)	(24-25)	And Not (26)	(27-28)	And Not (29)	(30-31)
01	O	OUTPUT	H				Ø6			OF				
02	O	O	R							1P				
03	O		D			1			N	1Ø	N	2Ø		3Ø
04	O	A	N	D						25				
05	O		D			1				2Ø				
06	O	O	R							3Ø				
07	O		T		2					L2	N	MR		
08	O	TOTALREC	T			1				L2		MR		
09	O	TOTCARD	T	2						L2				
10	O													
11	O													
12	O													

Figure 22.2 Indicators to condition output records.

01. The printer carriage skips to line 6 before printing. The heading is printed only when an overflow or first page condition occurs.

03. A detail line is printed when indicators 10 and 20 are off and indicators 25 and 30 are on. The indicators could have been previously specified as field indicators from input specifications or as resulting indicators from calculation specifications.

05. A detail line is printed if indicator 20 or 30 is on.

07. Two lines are spaced before printing a total if level 2 indicator is on and the matching record and halt 2 indicators are off. Thus, a total is taken after a control break, if no error has occurred. (It is assumed an error would turn the halt indicator on.)

08. A total is printed if control level indicator L2 is on and the matching record indicator is also on.

09. A summary card is punched and placed in stacker 2 if control level indicator L2 is on.

Columns 32 through 37 are used to enter the name of every field that is to be written out. Names must have been previously assigned on the preceding forms.

Column 38 provides an entry to edit numerical fields produced on printed reports. That is, the field may be punctuated by automatically adding commas and decimal points in the proper positions. Insignificant zeros to the left of the field may be suppressed as required. The various codes are printed on the form for ready reference. Figure 22.3 is a table which further explains the available editing codes.

For example, if code 1 is specified in column 38, the field will be printed with commas and a decimal point but with no sign,

Code	Commas	Decimal Point	No Sign	Signs for Negative Balance		Print Out on Zero Balance	Zero Suppress
				CR	Minus (-)		
1	X	X	X			.00	X
2	X	X	X			Blanks	X
3		X	X			.00	X
4		X	X			Blanks	X
A	X	X		X		.00	X
B	X	X		X		Blanks	X
C		X		X		.00	X
D		X		X		Blanks	X
J	X	X			X	.00	X
K	X	X			X	Blanks	X
L		X			X	.00	X
M		X			X	Blanks	X
X *							
Y **							X
Z			X				X

Note: * Code X moves a positive sign from the units position.

** Code Y puts slashes in a numeric field according to the following pattern:

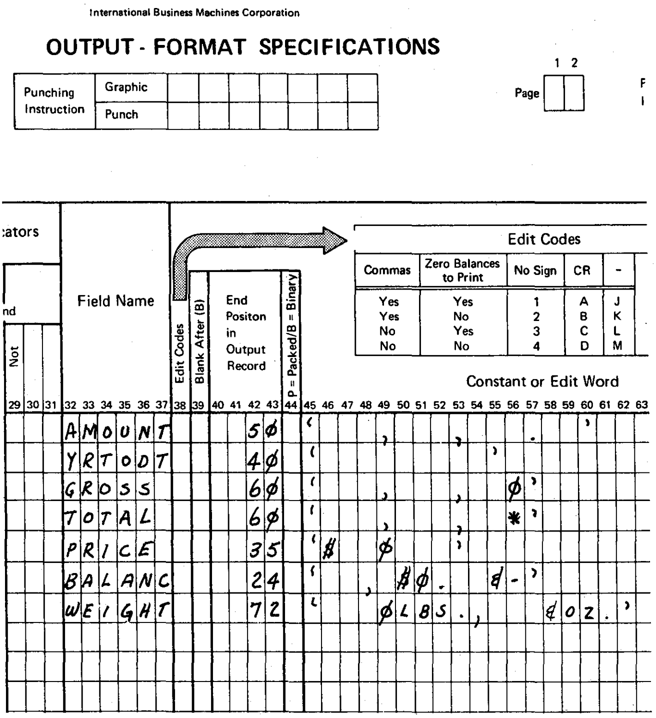
nn/n
nn/nn
nn/nn/n
nn/nn/nn

Code Y zero-suppresses only the left-most portion.

Figure 22.3 Edit codes.

even if the field is minus. In case the field is all zeros, two zeros will be printed after the decimal point as shown. The code A will produce the same editing as code 1 with the exception that the credit symbol (CR) will be printed after minus fields. If code J is used the minus symbol (—) rather than the credit symbol will be printed after minus quantities.

Columns 45 through 70 may be used for more elaborate editing of fields by inserting the desired symbol in the proper position of the field. The use of edit words is shown in Figure 22.4. Constants may also be entered in these columns. These are usually words used for report headings, column headings, and card identification. Structure of the constant must conform to the same rules as previously stated, and the word must be enclosed in apostrophes as shown.



International Business Machines Corporation

OUTPUT - FORMAT SPECIFICATIONS

Punching Instruction	Graphic							
	Punch							

Page 1 2

Edit Codes

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

Field Name (32–37)	Edit Codes (38)	Blank After (B) (39)	End Positon in Output Record (40–43)	P = Packed/B = Binary (44)	Constant or Edit Word (45–63)
AMOUNT			5ϕ		' , , . '
YRTODT			4ϕ		' '
GROSS			6ϕ		' , , ϕ'
TOTAL			6ϕ		' , , *'
PRICE			35		'ƀ ϕ '
BALANC			24		' ,ƀϕ. &-'
WEIGHT			72		' ϕLBS., &OZ.'

Unedited	*Edited*
24739000741	247,390,007.41
000743261	743261
000001421	1,421
000476222	***476,222
00002	$ .02
724325	$7,243.25 –
006015	6lbs., 015 oz

Figure 22.4 Use of edit words.

Column 39 is used to reset a field to zeros or to blanks. The feature is useful when totals are accumulated and printed for a control group. After a total is printed for one group, the field can be reset to zero. The field is then clear for accumulation of the next group.

Entry	*Explanation*
Blank	Field is not to be reset.
B	Field specified in columns 32 through 37 is to be reset after the output operation is complete. Numerical fields are reset to zeros, alphameric fields to blanks.

Columns 40 through 43 are used to indicate the location within the output record of the field that is to be printed. The number entered in the columns is the location of the units position of the field either on the printer or card.

The largest number that can be used to indicate the location of a card field is 96. The largest number for printed output is 128 or the capacity of the printer being used in the System/3.

The multifunction card unit can print as well as punch cards. When printing on cards is specified, an asterisk (*) is placed in column 40 before the number of the end printing position of the field.

Column 44 is not used.

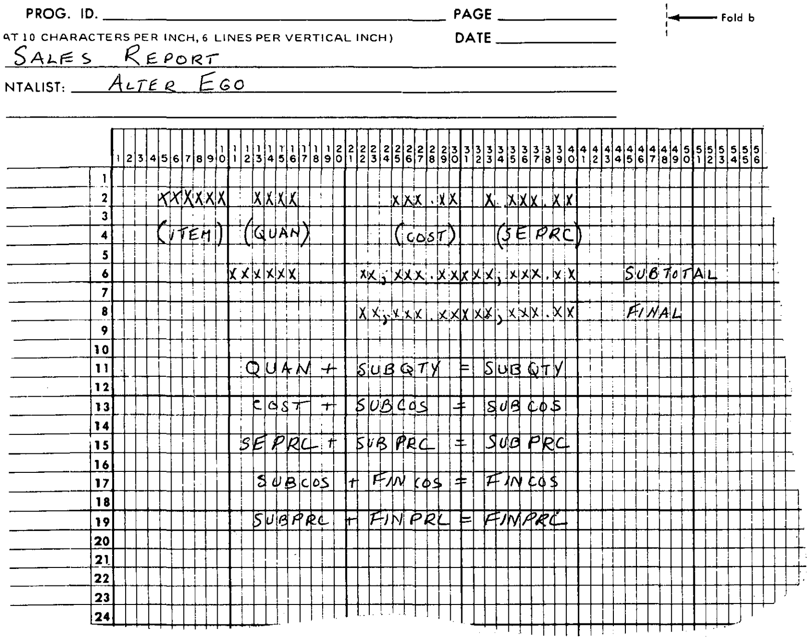
Example 1: Complete RPG II Program

The report in Figure 22.5 is to be produced showing item name, quantity, cost, and selling price. Subtotals of the quantity and value fields are to be accumulated by item with final totals at the end of the report. Input is a file of 96-column cards, previously sorted to item name.

```
Bolt        25      25.00      40.00
            60      60.00     100.00
            65      65.00     108.00

           150     150.00     248.00      Subtotal


Nut         10        .20        .50
          2000      40.00      60.00
            50       1.00       1.40
           180       3.60       5.00

          2240      44.80      66.90      Subtotal


Screw       40      80.00     100.00
            49      98.00     110.00

            89     178.00     210.00      Subtotal

                   372.80     524.90      Final
```

Figure 22.5 Sales report.

The layout of the report is planned in advance to determine where each field will be printed. Printer spacing charts are available for this purpose (Figure 22.6). Graph paper of suitable size can also be used.



PROG. ID. ______ PAGE ______ ← Fold b

AT 10 CHARACTERS PER INCH, 6 LINES PER VERTICAL INCH) DATE ______

SALES REPORT

NTALIST: ALTER EGO

```
             1111111111222222222233333333334444444444555555
    12345678901234567890123456789012345678901234567890123456
 1
 2      XXXXXX  XXXX        XXX.XX  X,XXX.XX
 3
 4      (ITEM)  (QUAN)      (COST)   (SE PRC)
 5
 6            XXXXXX     XX,XXX.XX XX,XXX.XX    SUBTOTAL
 7
 8                       XX,XXX.XX XX,XXX.XX    FINAL
 9
10
11              QUAN + SUBQTY = SUBQTY
12
13              COST + SUBCOS = SUBCOS
14
15             SEPRC + SUB PRC = SUB PRC
16
17              SUBCOS + FIN COS = FIN COS
18
19              SUBPRC + FIN PRC = FINPRC
20
21
22
23
24
```

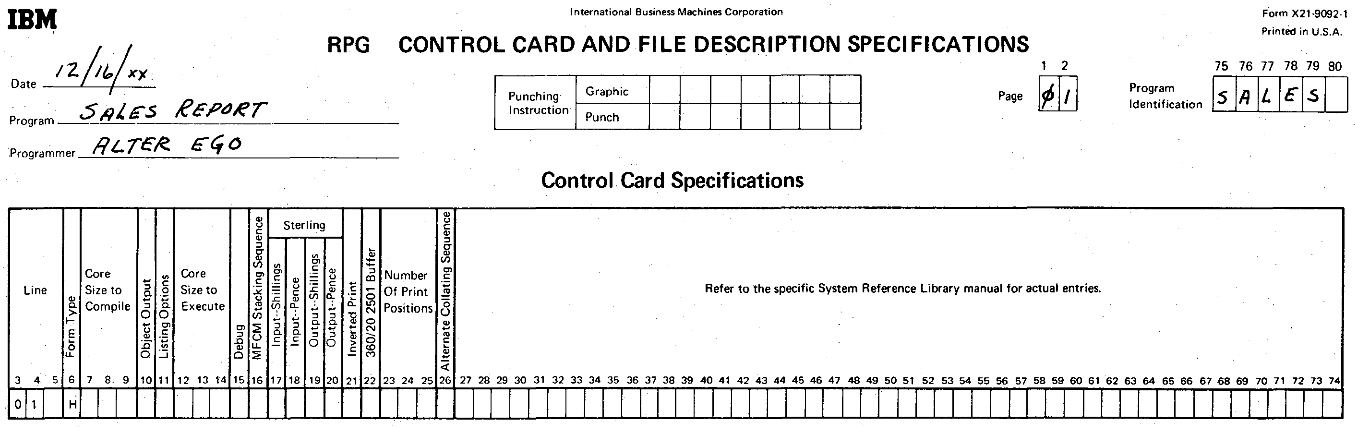
Figure 22.6 Sales report. Printer spacing chart.

Figure 22.7 is the Control Card Specification form showing that the program is to be assembled on an 8K System/3. No other entries are required.

Figure 22.8 describes the two files involved in the processing: the input card file ITEMS and the output file REPORT.

Figure 22.9 is the input form with each field of the input record named and located. The record identifying indicator is 01. Control level indicator L1 is assigned to the item field. The indicator will be turned on each time a control break occurs. Accumulation of the quantities, cost, and selling price occurs for every record. Totals are taken for each control break as conditioned by indicator L1 (Figure 22.10).

Figure 22.11 is the Output-Format Specification form. The detail line is printed as specified on lines 01 through 04. Note that edit code 3 is used to print the amount fields with commas and decimal points. The end position of each field corresponds with the location of the field shown on the printer spacing chart.



IBM

International Business Machines Corporation

Form X21-9092-1
Printed in U.S.A.

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

Date 12/16/xx

Program SALES REPORT

Programmer ALTER EGO

Punching Instruction								
	Graphic							
	Punch							

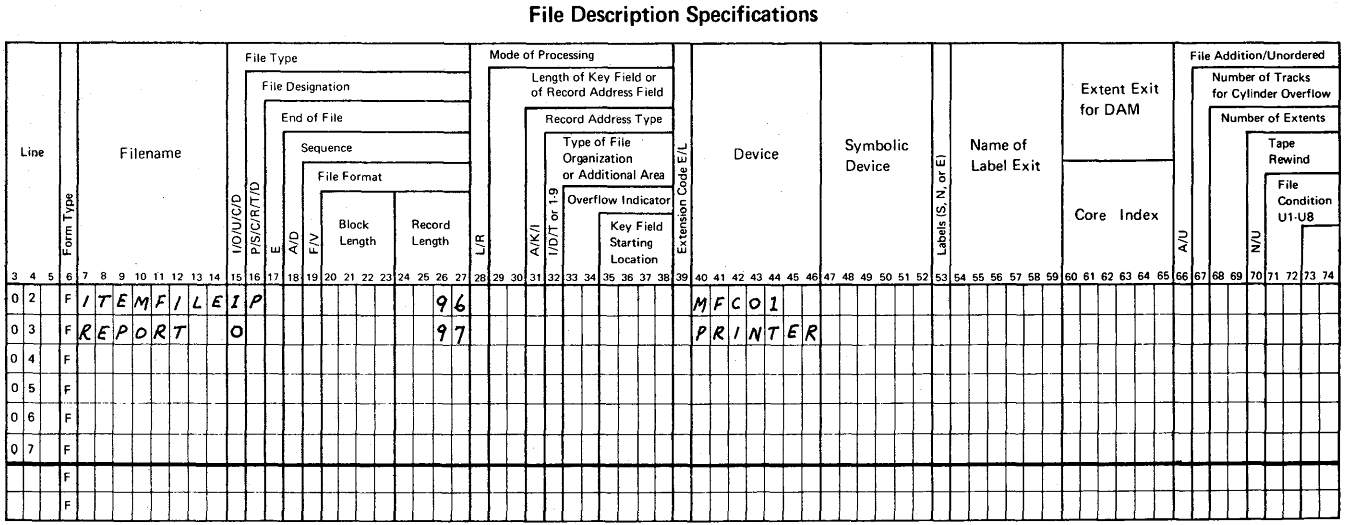
Page (1 2) Ø1

Program Identification (75 76 77 78 79 80) SALES

Control Card Specifications

Line	Form Type	Core Size to Compile	Object Output	Listing Options	Core Size to Execute	Debug	MFCM Stacking Sequence	Sterling: Input-Shillings	Sterling: Input-Pence	Sterling: Output-Shillings	Sterling: Output-Pence	Inverted Print	360/20 2501 Buffer	Number Of Print Positions	Alternate Collating Sequence	Refer to the specific System Reference Library manual for actual entries.
3 4 5	6	7 8 9	10	11	12 13 14	15	16	17	18	19	20	21	22	23 24 25	26	27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
0 1	H															

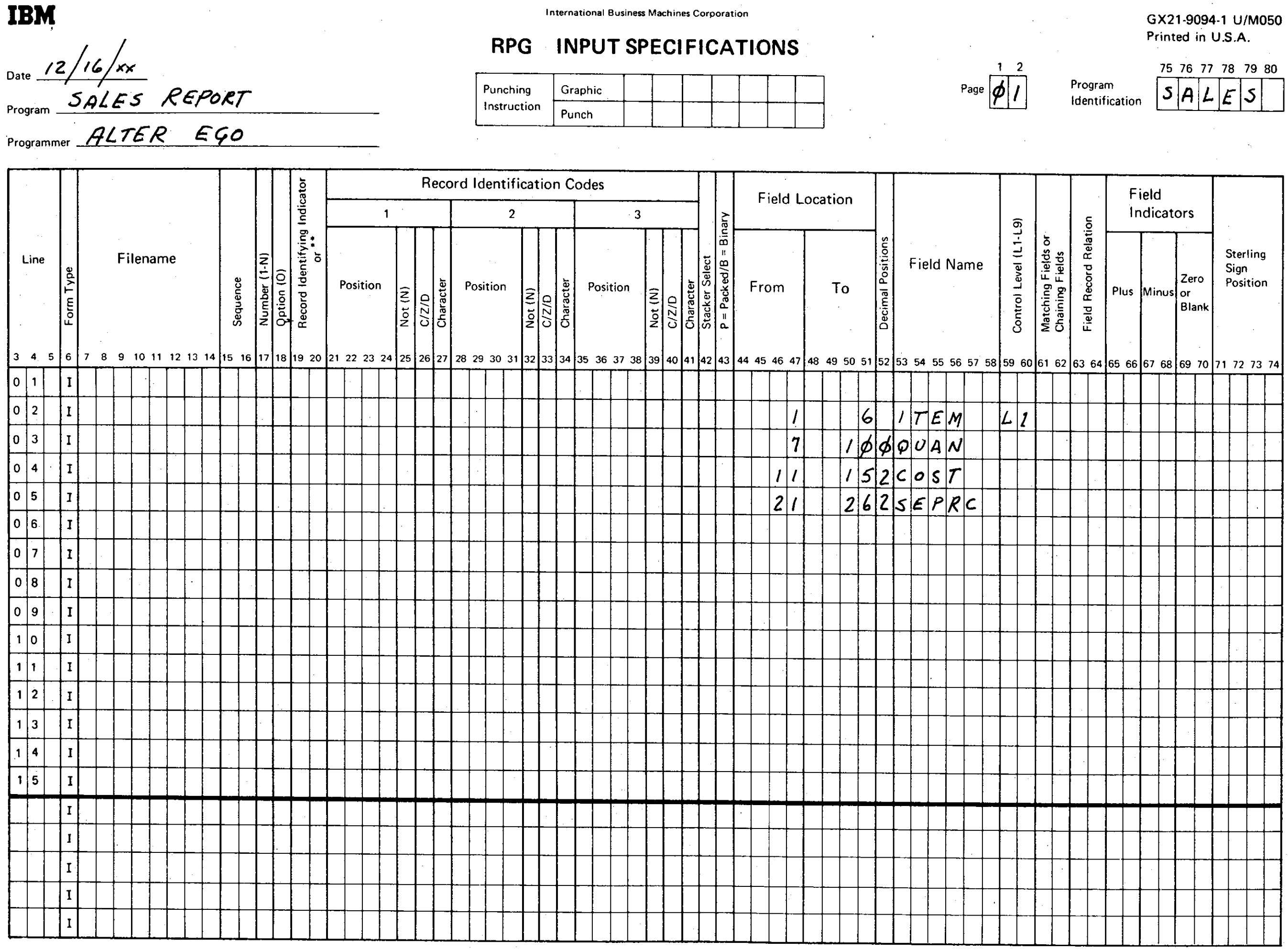
Figure 22.7 Sales report. Control card specifications.



File Description Specifications

Line	Form Type	Filename	File Type (I/O/U/C/D)	File Designation (P/S/C/R/T/D)	End of File (E)	Sequence (A/D)	File Format (F/V)	Block Length	Record Length	Mode of Processing (L/R)	Length of Key Field or of Record Address Field	Record Address Type (A/K/I)	Type of File Organization or Additional Area (I/D/T or 1-9)	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM / Core Index	File Addition/Unordered (A/U)	Number of Tracks for Cylinder Overflow	Number of Extents	Tape Rewind (N/U)	File Condition U1-U8	
3 4 5	6	7 8 9 10 11 12 13 14	15	16	17	18	19	20 21 22 23	24 25 26 27	28	29 30	31	32	33 34	35 36 37 38	39	40 41 42 43 44 45 46	47 48 49 50 51 52	53	54 55 56 57 58 59	60 61 62 63 64 65	66	67	68 69	70	71 72	73 74
0 2	F	ITEMFILE	I	P					96								MFC01										
0 3	F	REPORT	O						97								PRINTER										
0 4	F																										
0 5	F																										
0 6	F																										
0 7	F																										
	F																										
	F																										

Figure 22.8 Sales report. File description specifications.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 12/16/xx

Program SALES REPORT

Programmer ALTER EGO

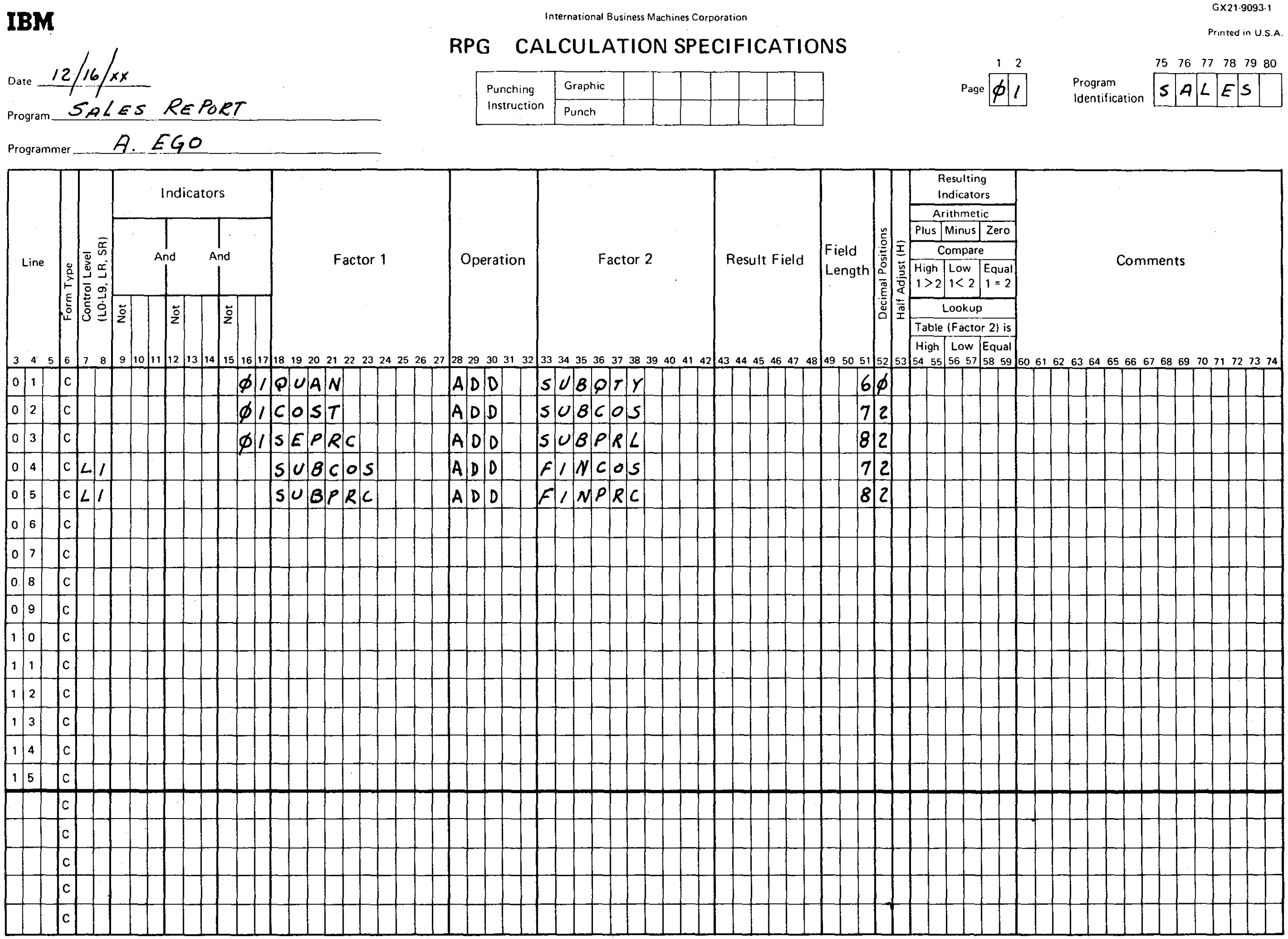
Punching Instruction	Graphic							
	Punch							

Page (1 2) Ø1

Program Identification (75 76 77 78 79 80) SALES

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes												Stacker Select	P = Packed/B = Binary	Field Location		Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators			Sterling Sign Position
							1				2				3																
							Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character	Position	Not (N)	C/Z/D	Character			From	To						Plus	Minus	Zero or Blank	
3 4 5	6	7 8 9 10 11 12 13 14	15 16	17	18	19 20	21 22 23 24	25	26	27	28 29 30 31	32	33	34	35 36 37 38	39	40	41	42	43	44 45 46 47	48 49 50 51	52	53 54 55 56 57 58	59 60	61 62	63 64	65 66	67 68	69 70	71 72 73 74
01	I																														
02	I																				1	6		ITEM	L1						
03	I																				7	1Ø	Ø	QUAN							
04	I																				11	15	2	COST							
05	I																				21	26	2	SEPRC							
06	I																														
07	I																														
08	I																														
09	I																														
10	I																														
11	I																														
12	I																														
13	I																														
14	I																														
15	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 22.9 Sales report. Input specifications.



IBM

International Business Machines Corporation

GX21-9093-1

Printed in U.S.A.

RPG CALCULATION SPECIFICATIONS

Date 12/16/xx

Program SALES REPORT

Programmer A. EGO

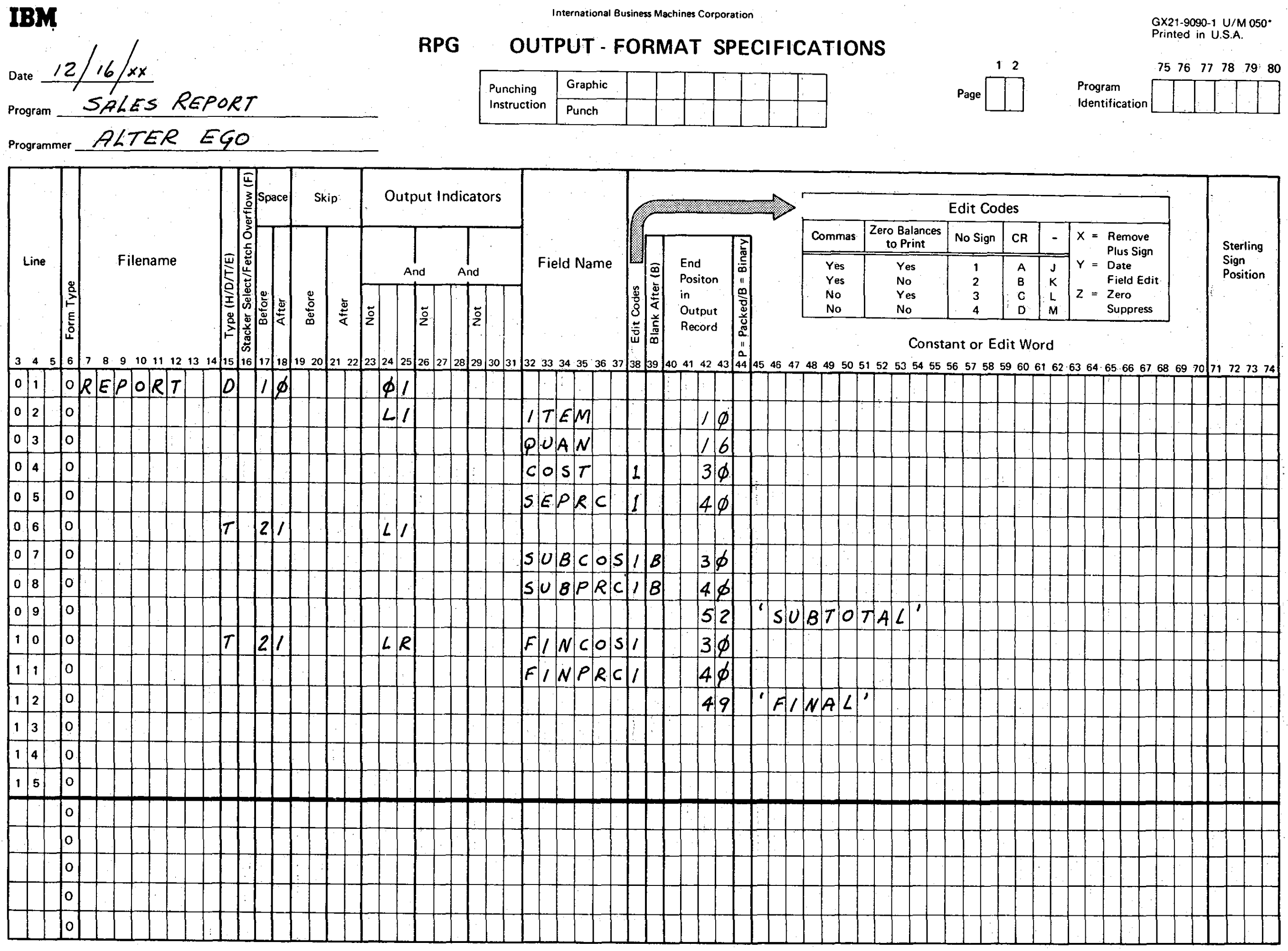
Punching Instruction: Graphic / Punch

Page (1 2) Ø1

Program Identification (75–80) SALES

Line (3–5)	Form Type (6)	Control Level (L0-L9, LR, SR) (7–8)	Indicators (9–17)	Factor 1 (18–27)	Operation (28–32)	Factor 2 (33–42)	Result Field (43–48)	Field Length (49–51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators (54–59)	Comments (60–74)
01	C		Ø1	QUAN	ADD	SUBQTY		6	Ø			
02	C		Ø1	COST	ADD	SUBCOS		7	2			
03	C		Ø1	SEPRC	ADD	SUBPRL		8	2			
04	C	L1		SUBCOS	ADD	FINCOS		7	2			
05	C	L1		SUBPRC	ADD	FINPRC		8	2			
06	C											
07	C											
08	C											
09	C											
10	C											
11	C											
12	C											
13	C											
14	C											
15	C											
	C											
	C											
	C											
	C											
	C											

Figure 22.10 Sales report. Calculation specifications.



IBM

International Business Machines Corporation

GX21-9090-1 U/M 050*
Printed in U.S.A.

RPG OUTPUT - FORMAT SPECIFICATIONS

Date 12/16/xx

Program SALES REPORT

Programmer ALTER EGO

Punching Instruction	Graphic							
	Punch							

Page (1 2) □□

Program Identification (75 76 77 78 79 80) □□□□□□

Edit Codes

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T/E) (15)	Stacker Select/Fetch Overflow (F) (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators (23-31)	Field Name (32-37)	Edit Codes (38)	Blank After (B) (39)	End Positon in Output Record (40-43)	P = Packed/B = Binary (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
01	O	REPORT	D		1	ϕ			ϕ1							
02	O								L1	ITEM			1ϕ			
03	O									QUAN			16			
04	O									COST	1		3ϕ			
05	O									SEPRC	1		4ϕ			
06	O		T		2	1			L1							
07	O									SUBCOS	1	B	3ϕ			
08	O									SUBPRC	1	B	4ϕ			
09	O												52		'SUBTOTAL'	
10	O		T		2	1			LR	FINCOS	1		3ϕ			
11	O									FINPRC	1		4ϕ			
12	O												49		'FINAL'	
13	O															
14	O															
15	O															
	O															
	O															
	O															
	O															
	O															

Figure 22.11 Sales report. Output-format specifications.

Subtotals are printed after each control break with the constants to identify the amounts as either subtotals or a final total. The fields are blanked after printing so that they can correctly accumulate quantities for the next control group.

The final totals are printed by conditioning with the last record indicator. Setting these fields back to zero is optional.

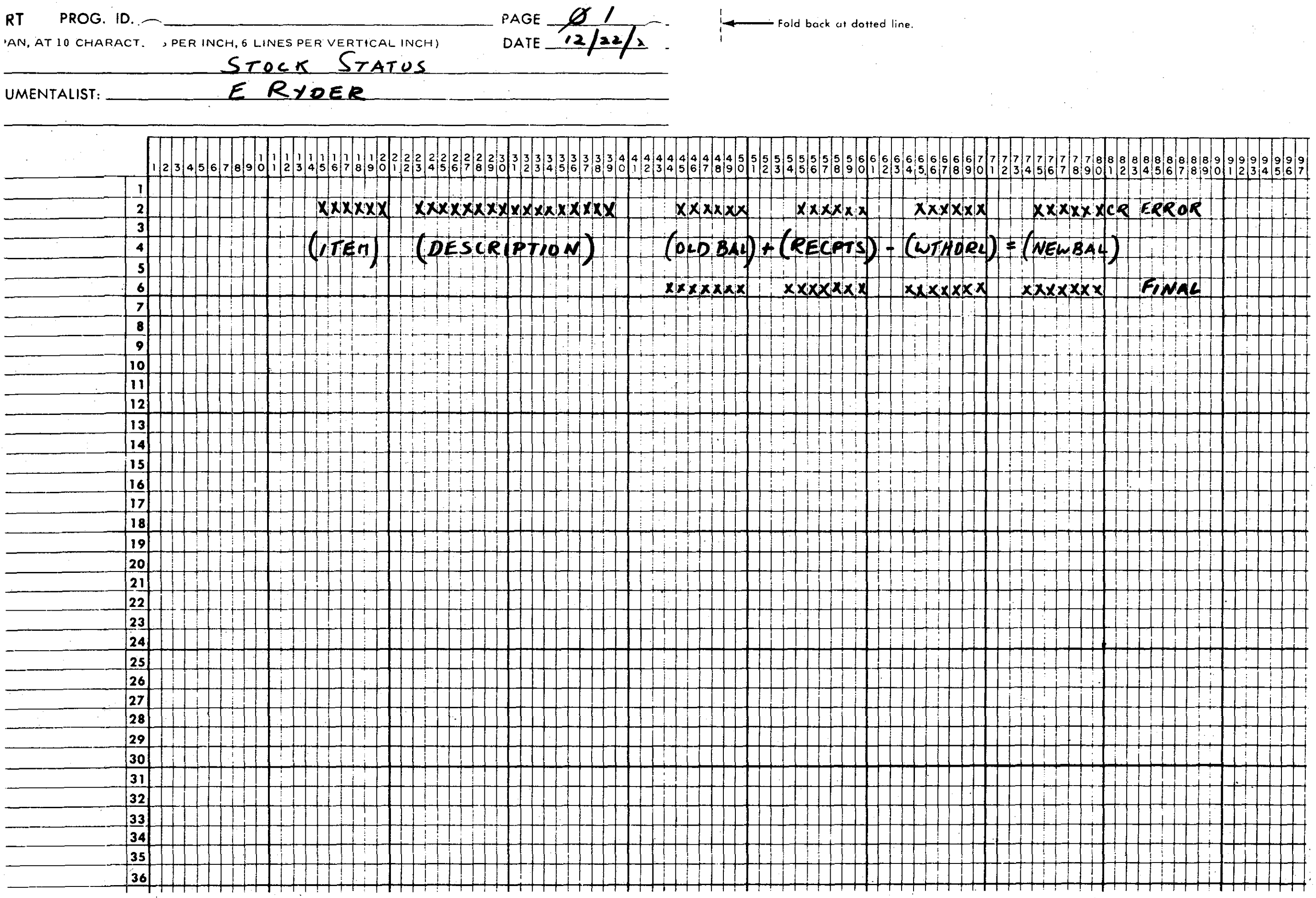
Example 2: Complete RPG II Program

A planned stock status report shown in Figure 22.12 is to be prepared from a file of inventory transaction cards. The report will include item number, item description, previous balance of the quantity in stock, withdrawals and receipts for the current period, and new balance in stock. As the report is printed, summary cards are punched with item number, description, and new stock balance.

Input records are the file of previous balance cards sorted to item number with current withdrawal and receipt cards. Figure 22.13 is a layout of the various types of cards. Figure 22.14 is the system flowchart of the overall procedure. The Control Card and File Description form is shown in Figure 22.15. All of these entries should now be familiar.

Each file is named and associated with an input or output device. Note that the old balance file contains three types of cards: previous balance, receipts, and withdrawals. These are described in the input specifications shown in Figure 22.16. Only one previous balance card is allowed in the file; the record is assigned indicator 01. More than one receipt or withdrawal is allowed because there may have been any number of transactions for an item number during the current period. Receipts are assigned indicator 02; withdrawals, indicator 03. Each type of card is identified by a unique punch in column 10: the digits 5, 6, and 8.

All three types of cards contain fields in the same card columns and are therefore associated with an OR in columns 14 and 15 of the form. The fields are located by their card columns, corresponding to the previous card layout form. Control level 01 is assigned to *item* because a total is to be printed and punched at the end of each group of cards with the same item number. The quantity field is assigned the halt indicator H1. If the input file contains a minus previous balance, an error is indicated and the program is stopped.



RT PROG. ID. — PAGE Ø1

'AN, AT 10 CHARACT. › PER INCH, 6 LINES PER VERTICAL INCH) DATE 12/22/x

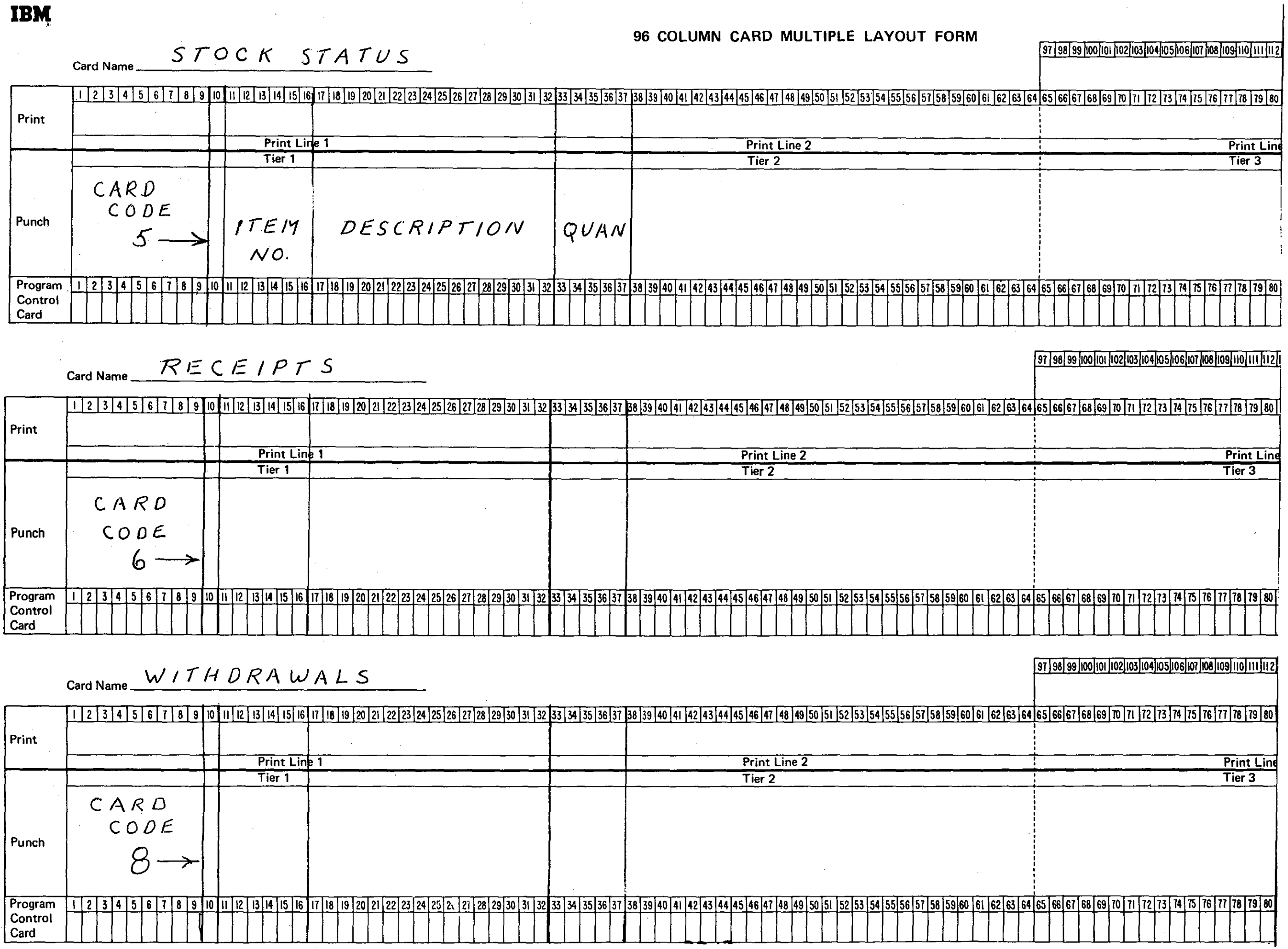
Fold back at dotted line.

STOCK STATUS

UMENTALIST: E RYDER

```
 1
 2               XXXXXX  XXXXXXXXXXXXXXXXXXXX  XXXXXX    XXXXXX    XXXXXX    XXXXXXCR ERROR
 3
 4              (ITEM)  (DESCRIPTION)        (OLD BAL) + (RECPTS) - (WTHDRL) = (NEWBAL)
 5
 6                                           XXXXXXX   XXXXXXX   XXXXXXX   XXXXXXX   FINAL
```

Figure 22.12 Stock status report. Printer spacing chart.



IBM

96 COLUMN CARD MULTIPLE LAYOUT FORM

Card Name STOCK STATUS

97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112

Print 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Print Line 1 Tier 1 — Print Line 2 Tier 2 — Print Line Tier 3

Punch CARD CODE 5 → ITEM NO. DESCRIPTION QUAN

Program Control Card 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Card Name RECEIPTS

97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112

Print 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Print Line 1 Tier 1 — Print Line 2 Tier 2 — Print Line Tier 3

Punch CARD CODE 6 →

Program Control Card 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Card Name WITHDRAWALS

97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112

Print 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Print Line 1 Tier 1 — Print Line 2 Tier 2 — Print Line Tier 3

Punch CARD CODE 8 →

Program Control Card 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Figure 22.13 Stock status. Card layout.

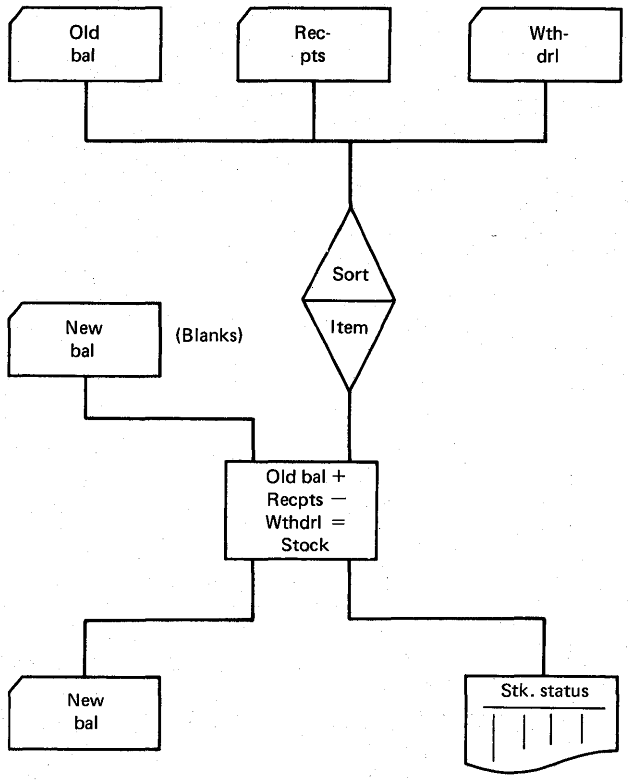


Figure 22.14 Stock status. System flowchart.

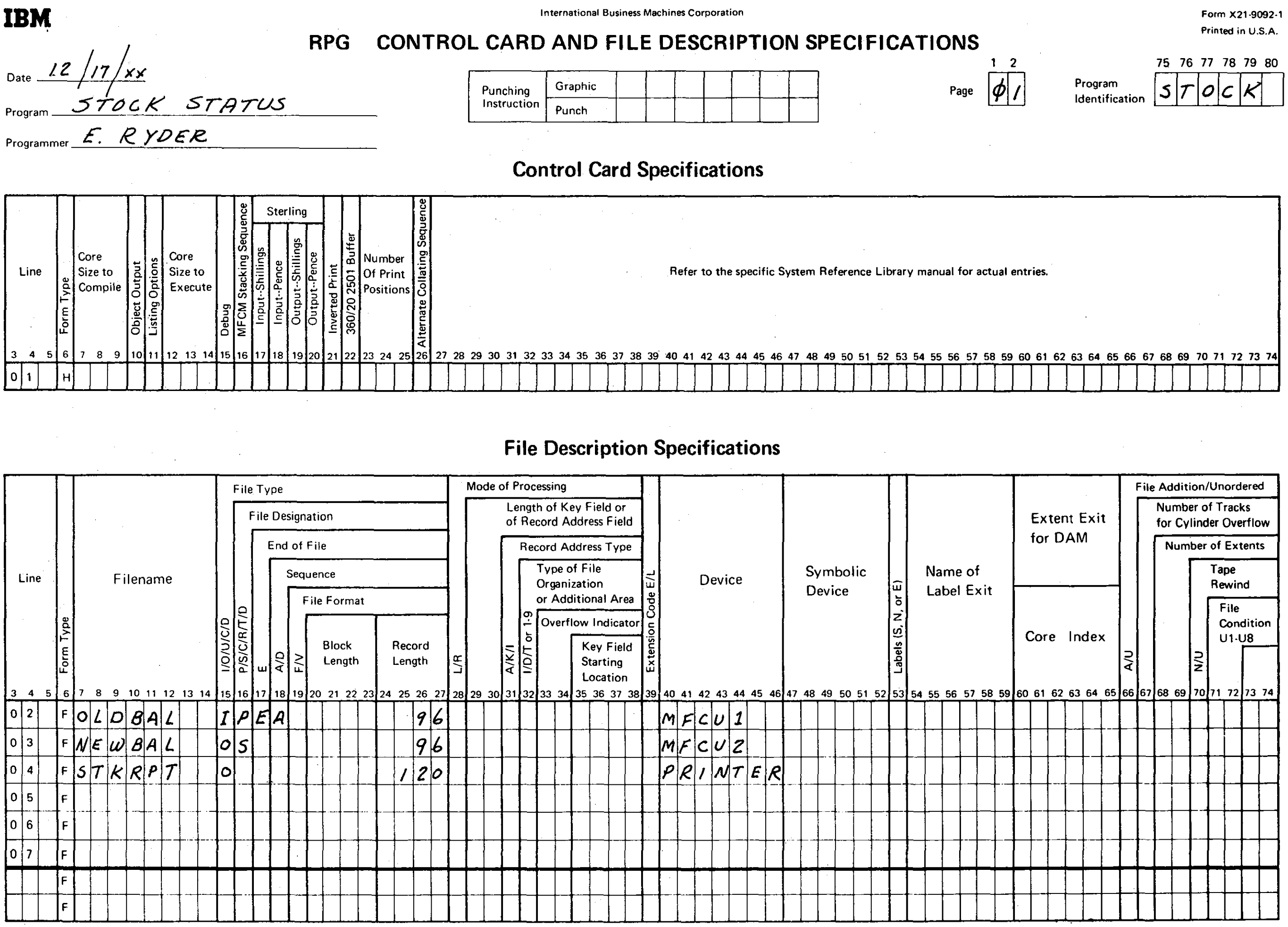
Figure 22.17 shows the calculation specifications. The following item numbers refer to the line numbers on the form.

01. Quantity from the previous balance card (indicator 01) is placed in the old balance field in computer storage. The field has been cleared after printing totals from the previous control group.

02. Quantity from the previous balance card is also placed in the new stock balance field.

03 and 04. Quantity from the receipt card (indicator 02) is placed in the receipts field in storage and is added to the stock field.

05 and 06. Quantity from the withdrawal card (indicator 03) is added to the withdrawal field and subtracted from the stock field. Indicator 09 is turned on if the field becomes minus.



IBM

International Business Machines Corporation

Form X21-9092-1
Printed in U.S.A.

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

Date 12/17/xx

Program STOCK STATUS

Programmer E. RYDER

Punching Instruction	
Graphic	
Punch	

Page (1 2) ϕ1

Program Identification (75 76 77 78 79 80) STOCK

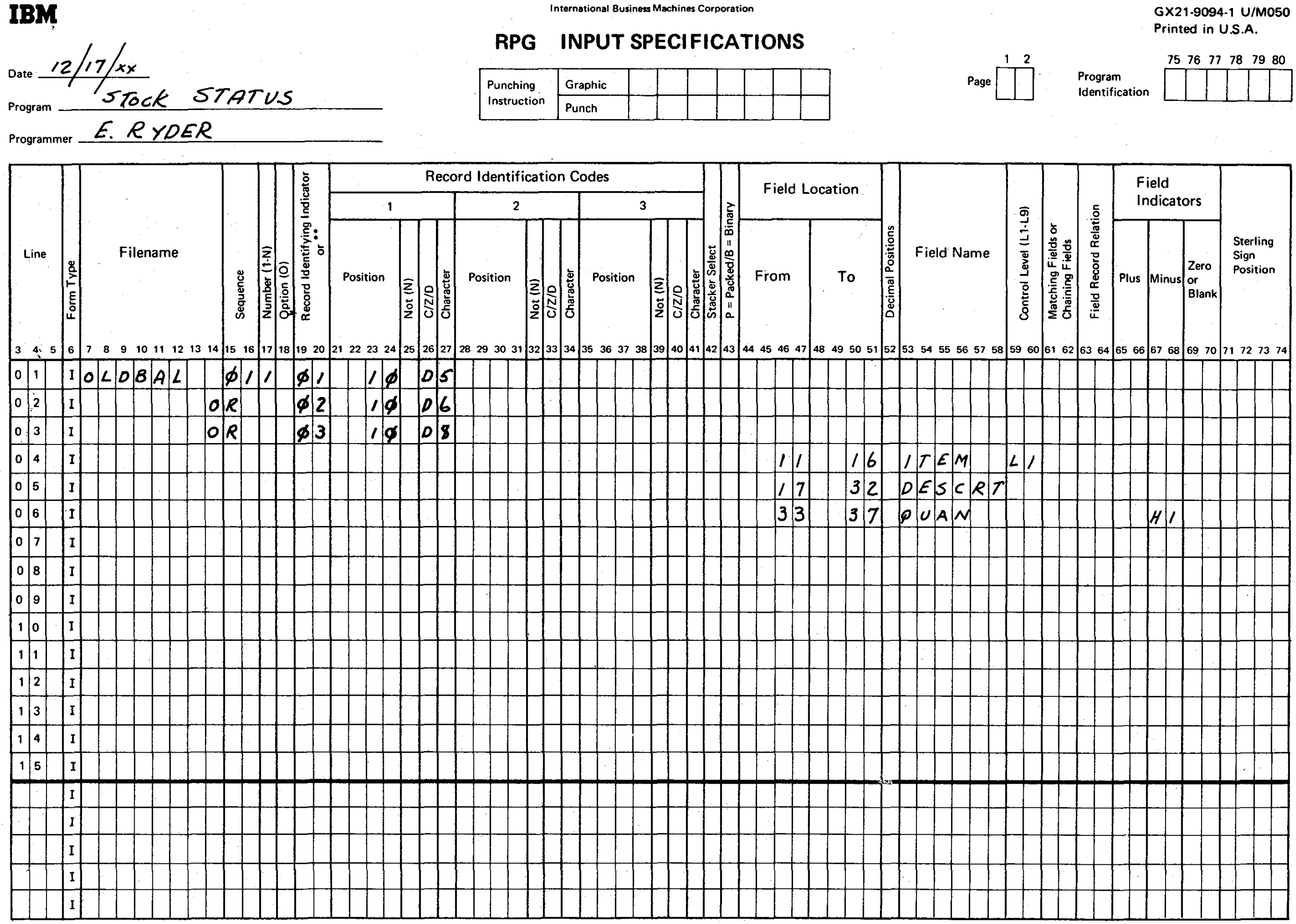
Control Card Specifications

Line (3-5)	Form Type (6)	Core Size to Compile (7-9)	Object Output (10)	Listing Options (11)	Core Size to Execute (12-14)	Debug (15)	MFCM Stacking Sequence (16)	Sterling: Input-Shillings (17)	Sterling: Input-Pence (18)	Sterling: Output-Shillings (19)	Sterling: Output-Pence (20)	Inverted Print (21)	360/20 2501 Buffer (22)	Number Of Print Positions (23-25)	Alternate Collating Sequence (26)	Refer to the specific System Reference Library manual for actual entries. (27-74)
0 1	H															

File Description Specifications

Line (3-5)	Form Type (6)	Filename (7-14)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D (16)	End of File E (17)	Sequence A/D (18)	File Format F/V (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing L/R (28)	Length of Key Field or of Record Address Field (29-30)	Record Address Type A/K/I (31)	Type of File Organization or Additional Area I/D/T or 1-9 (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)	Symbolic Device (47-52)	Labels (S, N, or E) (53)	Name of Label Exit (54-59)	Extent Exit for DAM Core Index (60-65)	File Addition/Unordered A/U (66)	Number of Tracks for Cylinder Overflow (67)	Number of Extents (68-69)	N/U (70)	Tape Rewind / File Condition U1-U8 (71-72)	(73-74)
0 2	F	OLDBAL	I	P	E	A			96								MFCU1										
0 3	F	NEWBAL	O	S					96								MFCU2										
0 4	F	STKRPT	O						120								PRINTER										
0 5	F																										
0 6	F																										
0 7	F																										
	F																										
	F																										

Figure 22.15 Stock status. Control card and file description.



IBM

International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-1 U/M050
Printed in U.S.A.

Date 12/17/xx

Program STOCK STATUS

Programmer E. RYDER

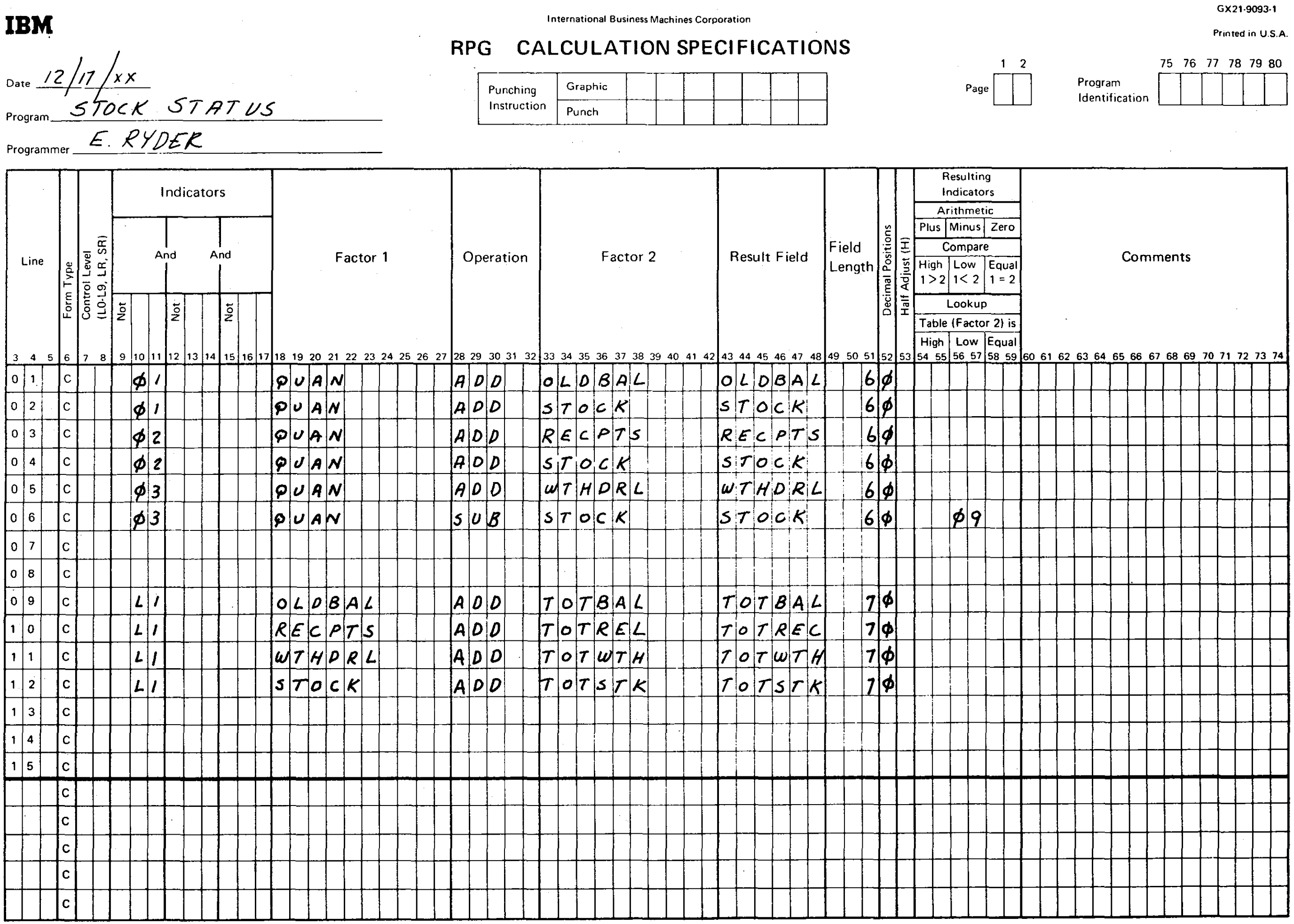
Punching Instruction	Graphic						
	Punch						

Page 1 2

Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes 1				2				3				Stacker Select	P = Packed/B = Binary	Field Location		Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators			Sterling Sign Position
3 4 5	6	7–14	15 16	17	18	19 20	Position 21–24	Not (N) 25	C/Z/D 26	Character 27	Position 28–31	Not (N) 32	C/Z/D 33	Character 34	Position 35–38	Not (N) 39	C/Z/D 40	Character 41	42	43	From 44–47	To 48–51	52	53–58	59 60	61 62	63 64	Plus 65 66	Minus 67 68	Zero or Blank 69 70	71–74
0 1	I	OLDBAL	Ø1	1		Ø1	1Ø		D	5																					
0 2	I	OR				Ø2	1Ø		D	6																					
0 3	I	OR				Ø3	1Ø		D	8																					
0 4	I																				11	16		ITEM	L1						
0 5	I																				17	32		DESCRT							
0 6	I																				33	37		QUAN					H1		
0 7	I																														
0 8	I																														
0 9	I																														
1 0	I																														
1 1	I																														
1 2	I																														
1 3	I																														
1 4	I																														
1 5	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

Figure 22.16 Stock status. Input specifications.



IBM

International Business Machines Corporation

GX21-9093-1

Printed in U.S.A.

RPG CALCULATION SPECIFICATIONS

Date 12/17/xx

Program STOCK STATUS

Programmer E. RYDER

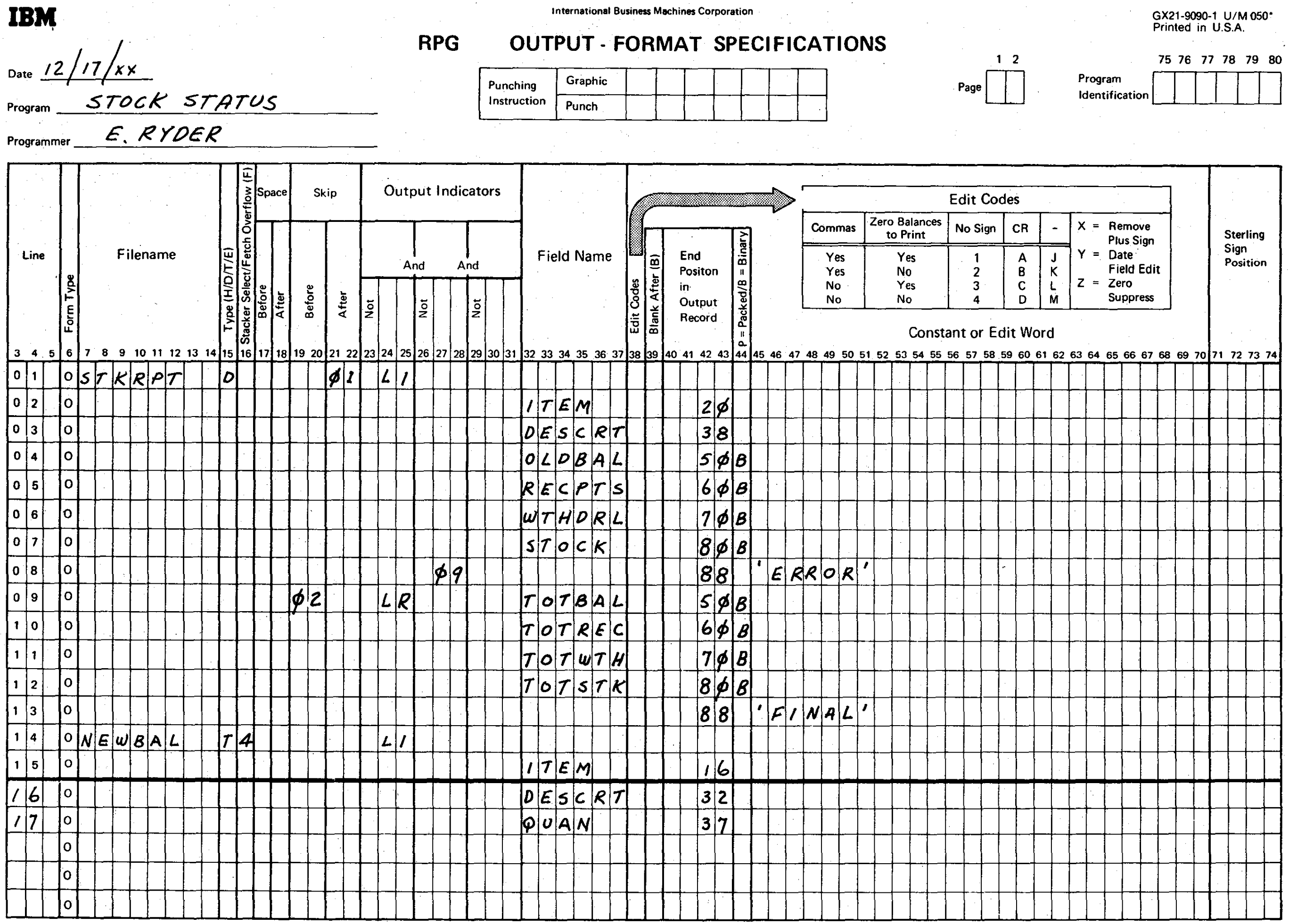
Punching Instruction								
	Graphic							
	Punch							

Page (1 2) □□

Program Identification (75 76 77 78 79 80) □□□□□□

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR) (7-8)	Indicators: Not (9)	(10-11)	And: Not (12)	(13-14)	And: Not (15)	(16-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Arithmetic Plus / Compare High 1>2 / Lookup Table (Factor 2) is High (54-55)	Minus / Low 1<2 / Low (56-57)	Zero / Equal 1=2 / Equal (58-59)	Comments (60-74)
01	C			Ø1					QUAN	ADD	OLDBAL	OLDBAL	6	Ø					
02	C			Ø1					QUAN	ADD	STOCK	STOCK	6	Ø					
03	C			Ø2					QUAN	ADD	RECPTS	RECPTS	6	Ø					
04	C			Ø2					QUAN	ADD	STOCK	STOCK	6	Ø					
05	C			Ø3					QUAN	ADD	WTHDRL	WTHDRL	6	Ø					
06	C			Ø3					QUAN	SUB	STOCK	STOCK	6	Ø			Ø9		
07	C																		
08	C																		
09	C			L1					OLDBAL	ADD	TOTBAL	TOTBAL	7	Ø					
10	C			L1					RECPTS	ADD	TOTREL	TOTREC	7	Ø					
11	C			L1					WTHDRL	ADD	TOTWTH	TOTWTH	7	Ø					
12	C			L1					STOCK	ADD	TOTSTK	TOTSTK	7	Ø					
13	C																		
14	C																		
15	C																		
	C																		
	C																		
	C																		
	C																		
	C																		

Figure 22.17 Stock status. Calculation specifications.



IBM

International Business Machines Corporation

GX21-9090-1 U/M 050*
Printed in U.S.A.

RPG OUTPUT - FORMAT SPECIFICATIONS

Date 12/17/xx

Program STOCK STATUS

Programmer E. RYDER

Punching Instruction	Graphic							
	Punch							

Page 1 2

Program Identification 75 76 77 78 79 80

Edit Codes

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T/E) (15)	Stacker Select/Fetch Overflow (F) (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23)	(24-25)	And Not (26)	(27-28)	And Not (29)	(30-31)	Field Name (32-37)	Edit Codes (38)	Blank After (B) (39)	End Positon in Output Record (40-43)	P = Packed/B = Binary (44)	Constant or Edit Word (45-70)	Sterling Sign Position (71-74)
01	O	STKRPT	D					ϕ1		L1											
02	O														ITEM			2ϕ			
03	O														DESCRT			38			
04	O														OLDBAL			5ϕ	B		
05	O														RECPTS			6ϕ	B		
06	O														WTHDRL			7ϕ	B		
07	O														STOCK			8ϕ	B		
08	O											ϕ9						88		'ERROR'	
09	O						ϕ2			LR					TOTBAL			5ϕ	B		
10	O														TOTREC			6ϕ	B		
11	O														TOTWTH			7ϕ	B		
12	O														TOTSTK			8ϕ	B		
13	O																	88		'FINAL'	
14	O	NEWBAL	T	4						L1											
15	O														ITEM			16			
16	O														DESCRT			32			
17	O														QUAN			37			
	O																				
	O																				
	O																				

Figure 22.18 Stock status. Output-format specifications.

(In planning any inventory application, it is necessary to anticipate the possibility of a minus balance in stock. The condition can occur when incorrect quantities of receipts and withdrawals are recorded.)

09, 10, 11, 12. When a control break in item number occurs, the balance just accumulated will be printed and punched as specified in the Output-Format Specifications. At the same time, the totals are added to fields when final totals are accumulated: total balance, total receipts, total withdrawals, and total stock.

Figure 22.18 shows the Output-Format Specifications. The two files are named the printed stock report and the punched stock card. The summary card produced in one run will be the previous balance card for the succeeding run.

One line is printed on the stock report whenever a control break occurs in item number. The fields named in lines 02 through 07 are printed. If indicator 09 is on, the minus balance is printed with the word ERROR. An output summary card is punched as specified in lines 14 through 17.

After the last record is processed from the input file, the final totals are printed as specified in lines 9 through 13. Totals may be crossfooted and balanced to control totals to check the accuracy of the run. With additional programming, the summary cards can also be totaled to prove that balances are carried forward to the next report accurately.

Questions and Exercises

1. Complete the Output-Format Specifications for all output files of the case study problem presented following chapter 20.
2. Prepare a review of your solution for class discussion.

INDEX

A and B registers, System/3, 351
Accountancy, 2
Accounting, 2
Accounting cycle, 10
Accounting entries, 40
Accounting formula, basic, 4, 6
Accounting machine, 241, 246-269, 298-308
Accounting period, 7
Accounting transactions, posted, 6, 7
Accounts payable, 5, 32, 33
  distribution reports, 33
  voucher, 33
Accounts receivable, 46-48, 118, 187
Accrual basis, accounting, 4
Accrued payroll, 29
Accumulate feature, IBM 129, 85
Accumulating totals, IBM 402, 259-263
Accumulator, 337
Accuracy:
  requirements for, 383
  transcription, 88
Acknowledge, 45
Actual cost, 29
Add, RPG II, 447
Add zoned decimal, RPG II, 358, 359
Adder, 338, 339
Addition, 341
Addition and Subtraction, IBM 604, 216
Address:
  disk track, 371
  storage, 333, 340
Address label writing, IBM 407, 299, 300
Address register, 337, 341
Advance, payroll, 29
All cycles, IBM 402, 255
Allocation of storage, 383
Allowances, 33
Alphabetic sorting switch, 150
Alternate track, disk storage, 373
American Standard Code for Information Interchange
    (ASCII), 333
Amount fields, 105
AND, RPG II, 415
Anticipation discount, 33
Applications processing programs, System/3,
    376, 377
Apron, 33
Arithmetic/logical unit, 31, 336, 337, 353
Arithmetic operations, RPG II, 446-455
Assembly, 386-388
  with optional manual control, 54
  phases of RPG II, 398
  program, 390
  with specifications, manual control, 54
Assembly run, 397
Asset, 2, 21
Asterisk, RPB II, 400
Attendance record, 25, 29, 101
Audit trail, 120
Automatic carriage, 245
Available, inventory, 40
Average cost, inventory, 40

Back order, 45
Back order control, 43
Balance card, 40
Balance sheet, 18, 21
Balance test for step suppression, IBM 604, 223
Base rate, 29
Basic conversion, 54
Basic operation, IBM 604, 227-239
Basic production, manual control, 53
Batch control, 132-134
Begin subroutine, RPG II, 464
Bill of lading, 43, 46
Billing, 42, 45, 46, 166, 191, 411, 412
  accounting entries, 45
Binary coded decimal (BCD), 329, 331

Binary coded decimal interchange code (EBCDIC), 350
Binary to decimal, fraction conversion, 323
Binary to hexadecimal, integer conversion, 324, 325
Binary notation, 316
Binary number systems, 315–317
Binary to octal:
  fraction conversion, 322
  integer conversion, 320
Binary State, 313, 314
Bonus, 29
Bookings reporting, System/3, 377
Bookkeeping, 1, 2
Bouchon, 58, 59
Branch instructions, 342
Branching operations, RPG II, 459–462
Brush, sorter, 138
Brushes, mark sensing, 210
Burden, 29
Burden account, 29
Buying service, file maintenance, 185, 186
Byte, 316
Byte capacity, RPG II, 402

Calculation, IBM 604, 215–218
Calculation capacity, IBM 604, 216, 217
Calculation results, 102
Calculation specifications, RPG II:
  columns 7,8, 436
  columns 9–17, 436, 438
  columns 18–27, 438
  columns 33–42, 438–440
  columns 43–48, 440, 441
  columns 49–51, 441
  column 52, 441
  column 53, 441
  columns 54–59, 442
Capital, 2
Card arrangement, RPG II source deck, 400
Card capability, System/3, 80-column, 361
Card capacity, 70
Card check, 102
Card counter:
  IBM 129, 85
  sorter, 163
Card data recorder:
  IBM 129, 79–86
  IBM 5496, 90–97
Card design:
  automatic punches, 110
  card punches, 109
  computers, 110, 111
  control panel wiring, 111
  machine characteristics, 109
  sorters, 110
Card feeds, System/3, 344
Card fields, 70, 71
Card handling:
  manual, 112
  sorter, 144–146
Card input, System/3, 344, 345
Card-oriented programs, System/3, 363, 364
Card output, System/3, 346–349
Card read punch, IBM 521, 218, 219
Card reading, IBM 402, 247, 248
Card selection, sorter, 152, 153
Card speeds, System/3, 345

Cash basis, accounting, 4
Cash disbursement register, 33
Central processing unit (CPU), 311, 336, 337
Chain printer, System/3, 348, 349
Channel, System/3, 351
Character emitter, IBM 407, 306
Character printing, System/3, 347, 348
Characters, special, 136
Chart of accounts, 11, 21
Check bytes, disk storage, 372
Check digit, 125, 126
Checking filing of detail behind master cards, 190
Checking interspersed gangpunching, 199, 201
Chute blade, sorter, 139
Class selection, 284, 285
Clock number, 29
Clock station, 30
Closing out books, 17
Closing out income and expense, 9
COBOL, 393, 394
Code checking, 328, 329
Codes, one digit, 105
Collate, System/3, 364–366
Collating sequence, alteration of, 403
Collator:
  card paths, 168, 171
  IBM 88, 168–192
Column indicator, IBM 5496, 91
Columns:
  mark sensing, 210–212
  punched card, 63, 70, 71
Column shift, IBM 604, 220, 221
Column split, IBM 604, 236
Combined files, RPG II, 398
Combined reproducing and gangpunching, 206–208
Comment line, RPG II, 400, 401
Comments, program, 387
Common channel, IBM 604, 220
Compare, RPG II, 456–458
Compare and testing operations, 456–458
Comparing:
  IBM 402, 264, 265, 268
  IBM 519, 194, 205
  two fields in same card, 187, 188
Comparing unit:
  collator, 175
  IBM 519, 202, 203
Compilers, 391–393
Computer-programmer relationship, 385, 388–391
Computer system, 311
Condition register, System/3, 351–353, 358, 360
Console, 312, 338
Constant factors, 335
Constants, RPG II, 439
Contamination control, disk storage, 370
Continuous job card, 30
Control break, 421
Control field, 105, 167, 171, 178, 181, 190, 191, 374, 421
Control or Header Card Specifications, RPG II:
  columns 7–9, 402
  column 10, 402
  column 11, 402
  columns 12–14, 402
  column 15, 403
  columns 17–20, 403
  column 21, 403

Control or Header Card Specifications, RPG II (*cont.*):
  column 26, 403
Control level indicators, RPG II, 421, 422, 436, 438
Control panel, 171, 195, 204, 216, 218
  IBM 402, 249–252
  IBM 521, 227
Control panel wiring, IBM 402, 252–269
Control section, 311, 336
Control tape, IBM 407, 303, 304
Control total, 121, 123
Conversion program, System/3, 80- to 96-column
    cards, 368
Corner cuts, 65, 116
Co-selectors, 273, 282, 284
Cost accounting, 50–52
Counter:
  counter control, IBM 407, 271–277
  counter read-in minus, IBM 604, 223
  counter read-in plus, IBM 604, 223
  counter read-out, IBM 604, 223
  counter read-out and reset, 223
  IBM 402, 259–263
  IBM 604, 221, 225
Coupling exit, IBM 407, 274
Credit, 21, 33
Credit entry, 7, 15, 18
Credit memoranda, 45, 46
Crossfoot, RPG II, 452
Crossfooting:
  accounting machines, 286, 287
  IBM 604, 227–230
Current earnings card, 30
Cycle, System/3, 353
Cycle inventory taking, 40
Cycle stealing, System/3, 354, 355
Cylinder, disk storage, 371, 372

Daily time card, 30
Data:
  constant, 102
  determining card, 102, 103
  historical, 78, 87
  input, 77
  location in other cards, 107
  machine produced, 103
  real-time, 77
  reference, 103
  representation for machines, 60–63, 313–315
  symbolic representation, 61, 62
Data checking, System/3, 344
Data conversion, System/3, 102
Data entry keyboard, System/3, 361, 366
Data fields, disk storage, 372
Data format, System/3, 350, 351
Data output, 77
Data processing, 77
  department, 119
  procedure, 118
  system, 77, 310, 311
Data protection, disk storage, 370
Data recorder IBM 5496, 90–97
Data recording program, System/3, 366
Data sequence:
  determining, 107
  source document, 109
Data verification error, System/3, 367, 368
Data verifying program, System/3, 366–368
Day shift, 30
Debit entry, 7, 15, 18, 21, 33
Debit memo, 46
Debug, RPG II, 398, 467, 468
Debug operation, RPG II, 403
Debugging, 390, 391
Decimal to binary, integer conversion, 320
Decimal to hexadecimal, integer conversion, 326
Decimal to octal:
  fraction conversion, 322
  integer conversion, 319
Decimal positions, RPG II calculation, 441
Deduction authorization, 26
Deduction card, 30
Deduction register, 30
Defective track, disk storage, 372, 373
Denominating cash payroll, 30
Description, item, 46
Detail file, 167
Detail print, 241, 255–259
Digit selector:
  IBM 407, 280, 281
  IBM 604, 236
  sorter, 164
Direct access, disk storage, 374
Direct Labor, 28
  cost, 30
Disk cartridge, 369
Disk file organization, 373–375
Disk pack, 369
Disk recording, 369–370
Disk storage:
  operation, 369, 370
  organization, 370–373
  System/3, 368–375
Display panel, IBM 604, 221–223
Distribution of costs, 21, 30
Divide, RPG II, 449, 450
Division, IBM 604, 216, 224
Document register, 130–132
Double-punch and blank-column detection, 194, 197
Double transposition error, 126
Dual cards, 99, 100, 115
Dual programming feature, System/3, 355
Due date, 34
Duplication, IBM 129, 78

Earnings record, 30
Edit entries, RPG II, 474, 475
Edit stop switch, sorter, 154
Edit switch, sorter, 154
Editing, 186, 187
Electrographic pencil, 211
Electronic calculating punch, IBM 604, 220, 221
Employee number, 30
End-of-file condition, RPG II, 406
End printing, 80-column card, 194, 209, 210
End subroutine, RPG II, 464
Erasure, card records, 83, 90
Exception, RPG II, 467
Execute subroutine, RPG II, 464
Execution run, 398
Expense, 3, 7, 13
Extended binary coded decimal code (EBCDIC), 332
External indicator, RPG II, 408, 428, 438

Fabric requirements reporting, System/3, 377
Fabrication, 54
Factor storage, IBM 604, 220, 222, 225

Falcon, M, 58, 59
Fanning cards, 146
Feed station, IBM 129, 79
Field, card, 70, 88, 136, 142, 151, 152
Field, determining size, 103–106
Field, fixed length, 71
Field description, RPG II, 410
Field indicators, 431, 437
Field length, 71
  RPG II calculation, 446, 447
Field name, 440
Field positions, 71
Field record relation, 428
Field reference, 105
Field selection, IBM 407, 282
FIFO (first-in, first-out), 38–40
File, RPG II, 398
File and card type identification, RPG II, 410
File description, RPG II, 398
File description specifications, RPG II:
  columns 7–14, 405
  column 15, 405, 406
  column 17, 406
  column 18, 406
  columns 24–27, 406
  column 32, 407
  columns 33, 34, 407
  column 39, 407
  columns 40–46, 408
  columns 71, 72, 408
File feeding device, 137
File handling operations, miscellaneous, 184–192
File identification, RPG II, 404
File maintenance, 167–192
  summarization, 168
  transcription, 167
File name, RPG II, 405
Final total, IBM 402, 260
Finder card, 188
Finished goods requirements reporting, System/3, 377
Finished stock, 40
Fixed asset, 21
Fixed count check, 329
Fixed length records, disk storage, 372
Flag byte, 372
Floating dollar sign, 306
Flowchart, 381
Force, RPG II, 467
Forecasting, manual control, 55
Form design, IBM 407, 304–308
Form spacing, IBM 407, 302, 303
FORTRAN, 394, 395
Functional units, computer, 337, 338

Gang job card, 30
Gangpunched extensions, 30
Gangpunching, 181, 193–203, 205, 239
Garnishee, wage, 20
General ledger, 20
General register, 338
General storage, IBM 604, 220, 222, 225–227
Glossary:
  accounts payable terms, 33, 34
  accounts receivable, 49, 50
  cost accounting, 52, 53
  general accounting, 21, 22
  inventory, 40
  manufacturing, 56, 57

Glossary (*cont.*):
  order writing and billing, 45, 46
  payroll, 29
Go to, RPG II, 459
Grand total, IBM 402, 260
Graphic subsets, BCD code, 68, 69, 331
Gross earnings, 27, 30
Group bonus plan, 31
Group indication, 269
Group multiplication, 217
Group printing, 243, 421
Groups, card, 191

Half-adjust:
  IBM 604, 224
  RPG II, 441
Halt indicators, RPG II, 431, 437, 438, 443
Halt operation, 342
Hardware, 386
Heading information, 46
Hexadecimal to decimal, integer conversion, 325, 326
Hexadecimal number system, 317, 326
High-order character, 135, 136
Hollerith, Dr. Herman, 62, 63
Hopper, 116, 79, 90, 137, 158, 168, 174, 175
Housekeeping routines, 383, 384
Hubs, control panel, 251

IBM, 63
IBM card code, 64, 66, 67, 68
Identifier field, disk storage, 371, 372
Identifying characters, RPG II, 414–417
Idle time, 31
Income, 3, 13
Income statement, 21
Index, disk storage, 374
Indexed sequential access, disk storage, 374, 375
Indirect labor, 28, 31
Individual job card, 31
Input of combined file description, RPG II, 405
Input device, 77
Input files, RPG II, 398
  input file name, RPG II, 410
  input file record fields, RPG II, 417–421
Input/output channel, System/3, 354
Input specifications, RPG II:
  columns 15, 16, 410, 411
  column 17, 412
  column 18, 412
  columns 19,20, 412–414
  columns 21–41, 414
  column 42, 417
  columns 44–48, 417–421
  columns 59,60, 421
  columns 61, 62, 422
  columns 63, 64, 428
  columns 65–70, 428–431
Insignificant zeros, 71
Instruction counter, 340–342
Instruction execution, System/3, 353, 354
Instruction format, System/3, 356–360
Instruction register, 337
Instruction timing, 360
Instructions, 335, 336
Instructions as data 336, 354
Integer conversion, 319–323
Interleaving functions, System/3, 355
Intermediate total, 243, 260, 265, 287

Interpreting, punched cards, 102
Interrupt, 354, 355
Interspersed gangpunching, 197, 354, 355
Inventory, 34, 125
  average cost, 38
  costing finished goods, 38
  maintenance cost, 37
  obsolescence, 38
  order points, 36, 37
  perpetual, 35
  physical count, 34
  processing cost, 38
Inventory cards, 99
Inventory control, 34
Inventory formula, 35, 36
Inventory management, 38
Inventory requirements, 36
Inventory status file, 188
Invoice, 46
Invoice register, 34
Issues, 40

Jacquard, Joseph Marie, 59, 60
Job order, 31
Joggling cards, 145
Journal, 10, 21
Journalizing, 14

K bytes, 350
Key, disk record, 373, 374
Key numbers, 188, 189
Keyboard:
  IBM 129, 78, 81
Keypunching:
  80-column card, 78–84
  96-column card, 89

Labor distribution, 24, 26
Language:
  data representation, 315
  programming, 315, 397
Last record indicator, RPG II, 406, 436, 438
Layout, card forms, 107, 113, 115
Lead time, 40
Ledger, 21
Ledger accounts, 11
Level, language, 392
Liability, 3, 5, 21
Library, program, 383, 391, 392
LIFO (last-in, first-out), 38, 40
Line Counter Specifications, RPG II, 407, 471, 472
List, IBM 402, 256
List program, System/3, 96-column card, 364
Local storage registers, System/3, 353
Logical nature of computers, 315
Logical path, program, 342
Lookahead fields, RPG II, 413
Lookup, RPG II, 413
Low-order character, 135, 136

Machine cycle, 248, 339, 340
Machine oriented, 392
Manual cycle, 388
Manufacturing control, 53–56
Mark-sensed cards, 100, 116
Mark sensing, 194, 210–214
Master cards, 98
Master personnel cards, 31
Master rate card, 31
Matching fields, 422, 425
Matching record indicator, RPG II, 422
Matching two files, 185, 186
Material accounting, 34
Materials planning, 55
Memory:
  artificial, 335, 336
  machine, 60
Merging:
  with equal selection, 183–184
  merging two files, 175–178
  with selection, 178–183
  System/3, 365, 366
Microsecond, 339
Millisecond, 339
Minor total, 241, 242, 260, 265, 287
Miscellaneous data, 46
Mnemonics, 387
Modifier program, System/3, 377
Modulus, self-checking number, 126
Modulus 10, 127–129
Modulus 11, 129, 130
Move, RPG II, 452
Move left, RPG II, 453–455
Move remainder, RPG II, 450, 452
Move zone operations, RPG II, 455
MR Indicator, 438
Multifunction card unit, System/3, 344, 345
Multiple column selector, sorter, 155–157
Multiple line printing, IBM 407, 292–294
Multiple line read, IBM 407, 298–300
Multiple selection, IBM 407, 277–280
Multiplication, IBM 604, 216, 230–233
Multiplier-quotient storage, IBM 604, 220, 222
Multiply, 449
Multiply minus, IBM 604, 224
Multiply plus, IBM 604, 224

Nanosecond, 339
Net income, 4
Net loss, 4
Net pay, 27
Net worth, 6, 14, 18, 21
Normal alphamerical print entry, IBM 402, 256, 257
Normal zone entry, IBM 402, 258, 259
Numeric bits (BCD), 329, 331
Numerical print entry, IBM 402, 257

Object deck, 388
  RPG II, 402
Object program, 388, 390
Octal to binary:
  fraction conversion, 322
  integer conversion 320
Octal to decimal, integer conversion, 319, 320
Octal number system, 317, 318
Odd parity, 329, 331
Offset gangpunching, 201, 202
Offset gangpunching comparing, 202
Op code register, System/3, 353
Operands:
  instruction, 339, 340, 341, 387
  RPG II, 446
Operation codes:
  System/3, 353
  RPG II, 439, 446

Operation mnemonics, 387
Operation notes, IBM 604, 224
Operation part, instruction, 339, 340, 341
Operation planning, IBM 604, 223–227
Operations evaluation, manual control, 56
Operator controls, IBM 129, 80
OR, RPG II, 417, 418, 428
Order acknowledgment, 43
Order analysis, 43, 46
Order edit, System/3, 376, 377
Order writing, 42, 377
Output cards, 101
Output data, format, 383
Output device, 77
Output device assignment, RPG II, 408
Output files, RPG II, 398
Output-format specifications:
  columns 7–14, 469
  column 15, 469, 471
  column 16, 471
  columns 17, 18, 471
  columns 19–22, 471
  columns 23–31, 472
  columns 32–37, 474
  column 38, 474, 475
  column 39, 475, 476
  columns 40–43, 476
Output indicator, 472
Output stacker, RPG II, 471
Overflow indicators, RPG II, 438
Overflow skipping:
  IBM 407, 303
  System/3, 471

Paper forms, 349
  positioning, 245
Paper tape, 63
Parity bit, System/3, 350
Parity check, 329
  System/3, 351, 353
Payroll, 22, 118, 123, 124, 212, 214, 284, 285, 444, 465, 467
  file maintenance, 178–181
Payroll accounting entries, 28
Payroll advice, 31
Payroll change cards, 100
Payroll computation, 27
Payroll deduction, 31
Payroll master card, 31
Payroll record, master, 24, 25
Payroll register, 31
Payroll source documents, 24
Payroll writing, 28
Perpetual book inventory, 40
Personnel record, 25, 26
Physical count, 41
Pilot selector, IBM 407, 272, 274
Planning chart, IBM 604, 224
Plugwire, 251
Posting to accounts, 6, 7
Posting from journal, 14
Pricing file maintenance, 184
Primary sequence unit, collator, 171–174
Print chain, 349
Print checks, System/3, 349
Print wheels, IBM 407, 296, 297
Printer, System/3, 348, 349
Printer-keyboard, System/3, 360, 361, 385
Printing methods, 245
Problem analysis, 380, 381
Problem-oriented program, 392
Procedure design, 381
Procedure management, 121
Processing unit:
  System/3, 349
  System/3 data flow, 351–354
Processor check, RPG II, 397
Production statistics feature, IBM 129, 85
Profit and Loss (P & L), 17
Program assembly, 386–388
Program assembly checks, 390
Program control:
  IBM 129, 83, 84
  IBM 402, 263–265
Program execution, 340–342
Program generator, 397
Program indicators, RPG II, 428, 431
Program number, IBM 604, 224, 225
Program preparation, 381
Program routines, availability, 383
Program start, IBM 402, 265, 266
Program steps, 216
Program suppress, IBM 604, 225
Program unit, IBM 604, 221
Programmer, 379, 380, 391, 403, 404
Programming sheet, 387
Programming systems, 315, 380
Progressive total printing, IBM 407, 287
Property, Business, 4
Punch unit, IBM 519, 194, 195, 196
Punch X brushes, IBM 519, 197–202
Punched card:
  80-column specifications, 63
  96-column specifications, 72–75
  format, 87, 88
  types of, 98–102
Punching, 96-column card, 91
Punching position, 64
Punching station, IBM 129, 78, 80

Q register, System/3, 353

Random error, 126
Read check, System/3, 345
Read unit, IBM 519, 194
Read-units-into, out-of, IBM 604, 225
Read-X brushes, IBM 519, 204
Reading, IBM 407, 295
Reading station:
  IBM 129, 78
  IBM 402, 248, 249
  IBM 407, 295, 296
Receipts, 41
Reconciliation of bank statement, 31
Record count, 121
Record identifying indicator, RPG II, 412, 414, 437
Record keeping, 1
Record length, 406, 407
Records, historical, 166
Reference data availability, 383
Register, 337
Register masking, 338
Register shifting, 338
Reproduce/interpret program, System/3, 363, 364

Reproducing:
card, 193, 204–208
selective, 205, 206
Reproduction of data, 78, 79
Requisition, 41
Reservation, stock, 41
Resource management, 383
Restart, system, 383
Result field, RPG II, 441
Result indicators, RPG II, 434, 436, 437, 443, 444
Returns, 34
Revenue, 3
Right adjust, IBM 5496, 94
RPG II program:
example 1, 476, 477
example 2, 483–491
Row, card, 64, 169

Safety devices, disk storage, 370
Sales transaction, file maintenance, 181–183
Scheduling, manual control, 55
Scored cards, 102
Second reading, IBM 402, 258
Sectors, disk storage, 372
Selecting cards by key numbers, 188, 189
Selecting zero-balance cards, 189, 190
Selection switches, sorter, 152
Selector, 202–204
Self-checking number, 85, 124–130
Sequence:
alphabetic, 136
ascending, 135
descending, 135
intermediate, 136
major, 136
minor, 136
numerical, 135, 136
variable, 136
Sequence checking:
input files, 406
Sequencing, types of cards in file, 410, 412
Sequential access, disk storage, 373, 374
Set off, RPG II, 459
Set on, RPG II, 459
Ship to (invoice heading), 46
Shipping label, 43, 46
Shipping list, 42, 46
Short shipment, 34
Sight checking, 146
Sign control, IBM 604, 236, 237
Software, 386
Sold to (invoice heading), 46
Sort/collate program, System/3, 364–366
Sorter:
IBM 5486, 157–164
machine operation, 137–140
special devices, 154
Sorting:
96-column alphabetic, 160–163
96-column numerical, 159, 160
alphabetic, 149–152
alphabetic short cut, 153
block, 147–149
numerical, 142–144
printed reports, 140–142
Sorting needle, 146
Sorting time:
estimating, 146, 147
System/3, 366
Sorting tray, 159
Source deck, preparation of, 397
Source document, 46
Source program, 388, 390, 397
Special program, IBM 407, 288
Specification sheets, RPG II, 398–401
Square root, IBM 604, 237–239
SQRT, 392
Stacker:
collator, 168
IBM 129, 78, 80, 81
sorter, 138, 144
Standard assembly, manual control, 54
Standard BCD interchange code, 65, 69, 331
Standard cost, 31
Standard order quantity, 41
Status records, 166
Step-down, sequence, 171–174
Sterling currence, RPG II format, 403
Sticker, 34
Stock allocation, System/3, 377
Stock on hand, 41
Stock status, 45
Stock status report, 41
Storage, 333–335
computer, 336
IBM 129, 82, 83
System/3, 349–351
Storage address register, System/3, 351
Storage amount, program assembly, 386
Storage data register, System/3, 351
Storage register, 337
Storage units, IBM 407, 300, 301
Stored program, 335, 336, 379
Strings, records, 365
Subroutine operations, RPG II, 462–467
Subroutines, 342
Subtract, RPG II, 448
Subtract zoned decimal, 359, 360
Successive calculations, IBM 604, 233–236
Summarize, 241, 243
Summary punch, 168
Summary punching:
IBM 407, 288–292
IBM 519, 193, 194
Supervisory programs, 379
Suppress on minus balance, IBM 604, 223
Suppress on plus balance, IBM 604, 223
Synchronizer byte, disk storage, 372
System halts, RPG II, 402
System language, 388, 390

T account, 6, 7
Tables, RPG II, 407
Tabulate, 241
Tag, RPG II, 459–462
Tape-controlled carriage, IBM 407, 302, 303
Terms, 46
Test zone, RPG II, 458
Third reading, IBM 402, 257
Tiers, punched card, 73
Time card, 25
Time sharing, 354, 355
Time sheet, 31
Timekeeper, 26
Total program, IBM 402, 266, 267
Trade discount, 34

Transcript cards, 98
Transcription error, 125, 126
Transcription process, 87, 88
Transcription register, 41
Transmittal letter, 133
Transportation error, 126
Trial balance, 8, 16, 21, 34, 45
Type bar:
  alphanumeric, 246, 253
  numerical, 246, 253

Unit cost, 32
Unit count, 123, 124
Unit record, punched card, 64

Variable length overflow, IBM 407, 306
Variance, 32
Vaucanson, Jacques de, 58, 59
Vendor, 46
Verification, 96-column card, 94, 95, 96
Verification process, 88, 89
Voucher, 21

Warehouse system flowchart, 381
Work-in-process file, 32

X-punched cards, 191, 192, 217

Zero and add, RPG II, 447, 448
Zero balance, 189, 190
Zero check, IBM 604, 224
Zero elimination, sorter, 157
Zero and subtract, RPG II, 448, 449
Zero suppression, 307
Zone positions, BCD 329, 331
Zoned decimal format, 350