IBM®

Field Engineering
Manual of Instruction

IBM Confidential

This document contains information of a proprietary nature. ALL INFORMATION CONTAINED HEREIN SHALL BE KEPT IN CONFIDENCE. None of this information shall be divulged to persons other than IBM employees authorized by the nature of their duties to receive such information or individuals or organizations authorized by IBM in accordance with existing policy regarding release of company information. This manual has not been technically approved and is subject to change.

System/360 Model 20
2020 Processor

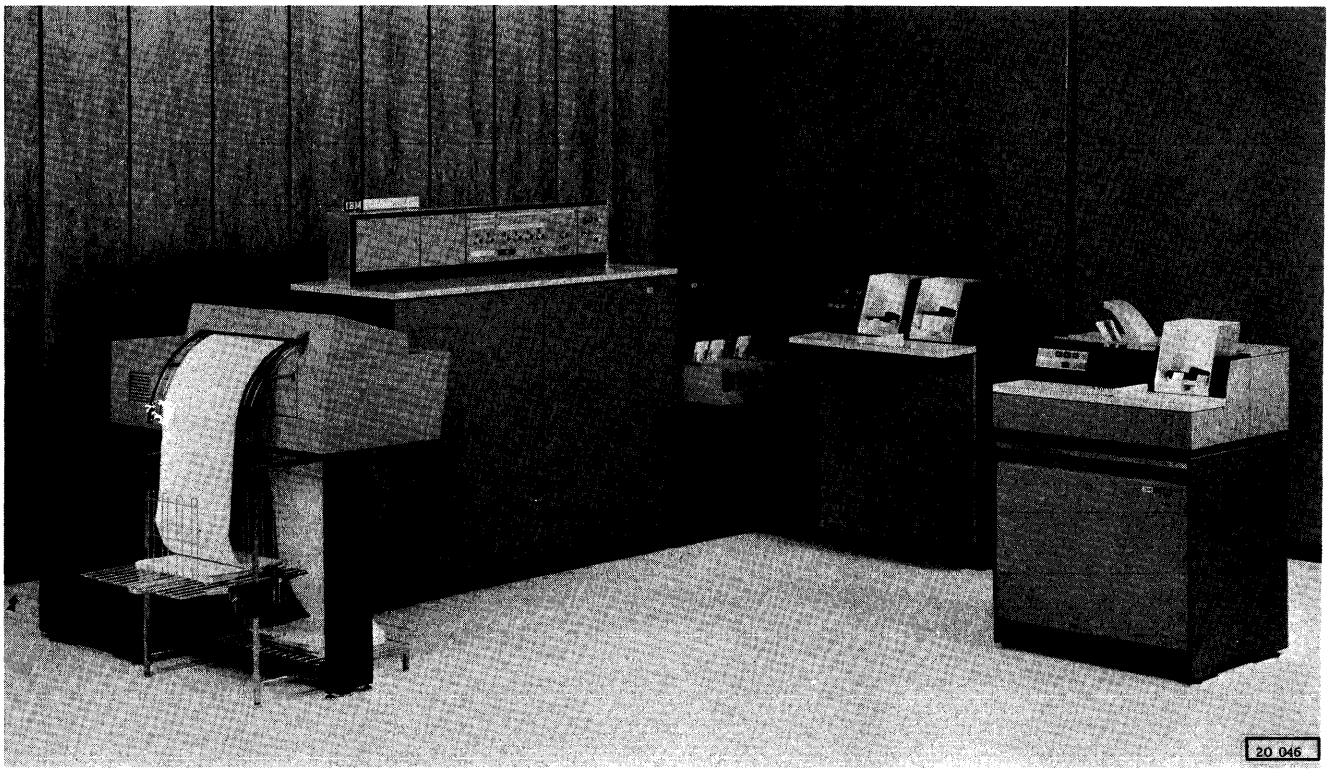
Revision Notice

This revision of the System/360 Model 20 FE Manual of Instruction makes the previous editions, form Z26-5909-1 and Z27-5909-0 obsolete.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.
Comments concerning the contents of this publication may be addressed to:
IBM, Product Publications Department, San Jose, Calif. 95114

SYSTEM/360 MODEL 20 FEMI

Figure List



IBM System/360 Model 20

SECTION 1 INTRODUCTION

1.1 SYSTEM INTRODUCTION

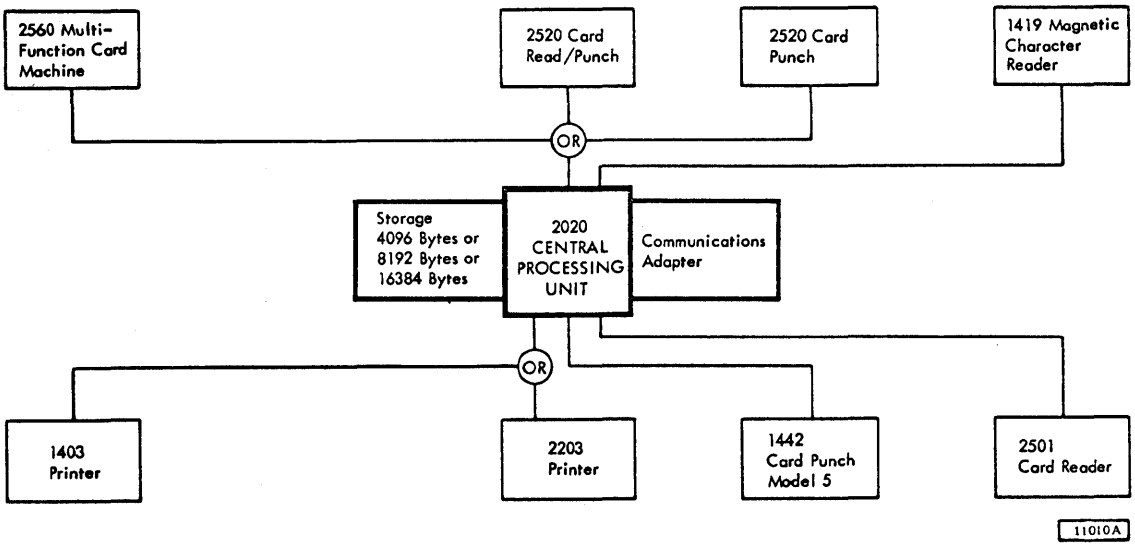
- The IBM System/360 Model 20 consists of a group of machine units operating together under control of a program.
- The machine program and the data to be operated upon are stored in a core storage area called Main Storage.
- Main Storage may have 4096, 8192, or 16384 addressable positions.
- Each addressable position contains 10 bits (one byte) of information.
- Thirty Seven machine instructions are available.
- Each machine instruction is executed by means of a series of micro-instruction steps.
- The micro-instruction steps used to perform machine instructions are stored in a storage area which can be read out only (Transformer Read Only Storage)
- Eight Data Registers and a modifier unit are provided for the manipulation of data.
- Input/Output units are operated under program control.
- An automatic interrupt feature enables the system to make optinum use of the I/O devices.
- Overlap of CPU and I/O operations allows the CPU to return to the main program after the transmission of each character from the I/O unit.

Description

The System/360 Model 20 Data Processing System consists of the IBM 2020 Processor, the IBM 1403 or 2203 Printer, the IBM 1442 Model 5 Punch, the IBM 2501 Card Reader, and the IBM 2520 Reader Punch or the IBM 2560 Multi-function Card Machine (Figure 1-1).

The CPU contains logic circuitry and two console panels, one for the customer and the other, normally covered, for the CE.

Data and instructions, entered into the system by way of a reader, are placed in Main Storage as logical data. Each of the positions of Main Storage can be addressed individually by a 12-bit binary address (4K Storage) and can store one byte of information. Main Storage addresses extend from 0000 to 4095 for 4K storage, 0000 to 8191 for 8K storage, and 00000 to 16383 for 16K storage. This addressing system enables selection of any byte or group of bytes within Main Storage.



The Model 20 can perform 37 different operations. An operation can be specified by a 2, 4, or 6 byte instruction which contains the operation code (Op Code) in the first byte. In the following bytes are one or two operand address as well as immediate data. Instructions are normally stored in consecutive locations in Main Storage and are executed sequentially. However, the sequence of operations may be altered at any point in the program by conditional branch instructions. Conditional branch instructions make logical decisions by performing tests on indicators or switches set by the computer or the operator. The System/360 Model 20 is an add-to-storage computer. That is, any position of Main Storage can be used for accumulation of a result. All executions of operations are accomplished by micro-programs which are stored in the TROS (Transformer Read-Only Storage) unit.

The Model 20 uses the System 360 machine language which is able to process fixed point data and variable length data.

In the Model 20, an automatic interrupt is provided to enable the system to make optimum use of the I/O devices, which signal the CPU at the end of a data transfer. And while the interrupt is independent of a programmed instruction, the interrupt feature can be disabled under program control.

In addition, the CPU is available for processing during most of each I/O operation, even though three separate devices may be functioning simultaneously. This overlap of I/O operations and CPU processing is called "time sharing" and is accomplished by allowing the CPU to return to the main program after transmission of each byte (character), while the designated I/O unit is completing the mechanical aspects of reading, punching, or printing. Time sharing provides the benefits of a buffer for I/O operations without sacrificing storage capacity or incurring the cost of a separate buffer.

The Model 20 CPU exerts direct control over all the I/O devices attached to it. However, I/O operations are initiated, halted, or tested by program instructions, which select the unit to be used. The I/O instructions determine what operation is performed (read, write, etc.) and where the data is stored.

1.2 INPUT-OUTPUT

1.2.1 Input Devices

- Input to the Model 20 CPU may take place through the IBM 2520 Card Read Punch, the IBM 2560 Multi-Function Card Machine, the IBM 2501 Card Reader or the IBM 1419 Magnetic Character Reader.
- Either the 2520 or the 2560 may be installed but not both.
- Input from externally located data transmission units as well as from other models of the IBM System/360 is possible by use of the Communication Adapter special feature.

IBM 2520 Card Read-Punch

The IBM 2520 Card Read-Punch provides punched-card input/output for the IBM System/360 Model 20. Cards are read, read and punched, or punched at the rate of 300 or 500 cards per minute (cpm) according to the model.

Model	Speed	Operation
A1	500 cpm	Read only, punch only, or read and punch simultaneously, as directed by the Model 20 program.
A2	500 cpm	Punch
A3	300 cpm	Punch

Card feeding is in parallel from the hopper, serial through the read station, and parallel through the punch station. Cards are ejected into radial stackers. The Model 20 program can select which of the two stackers receives each card. If no selection is made, the card goes into Stacker Number 1.

The IBM 2520 has lights and switches to provide the operator with information on operating conditions and control of start, stop, and card runout. The 2520 is under control of the Model 20 program during all phases of reading, punching, stacker selection, and data checking.

IBM 2560 Multi-Function Card Machine

The IBM 2560 Multi-Function Card Machine (MFCM) affords the IBM System/360 Model 20 a card-handling capability never before possible on a single pass through the system. With two hoppers, a solar-cell read station, a common punching station, an optional printing station, and five selective stackers, the MFCM offers full card-file maintenance abilities plus two, four, or six lines of card printing.

Cards from both the primary and secondary hopper can be read, punched, and selected into any one of the five stackers, regardless of the hopper of origin. The unit record functions of reproducing, gangpunching, summary punching, collating, decollating, and sorting can be performed on the MFCM under control of the Model 20 program. With the optional document printing feature, the MFCM functions include those of an interpreter and a punching, comparing card document printer.

IBM 2501 Card Reader

The IBM 2501 Card Reader provides punched-card input to the IBM System/360 Model 20. Cards are read at a maximum of 600 cards per minute (cpm) in the Model A1, and up to 1,000 cpm in the Model A2.

Card reading is accomplished by solar cells. Validity of the data is verified as each column is read. Each card is also checked for off-register punching and for incorrect positioning in the read station.

The 2501 functions under control of the Model 20 program. Lights and switches provide the operator with information on operating conditions and with control of start, stop, and card runout.

IBM 1419 Magnetic Character Reader

The 1419 Magnetic Character Reader can be attached to a System/360 Model 20 through the Serial Input/Output Channel device. The 1419 reads into the system the magnetically inscribed information on checks and other banking documents at speeds as high as 1600 documents per minute. Documents can be sorted into as many as 13 classifications as they are read.

The Serial Input/Output Channel, and in turn the 1419, operates under control of the Model 20 program.

Communications Adapter

The Communications Adapter special feature enables the Model 20 to function as a point-to-point data transmission terminal, operating under stored program control. This feature provides communication with the IBM 1009 Data Transmission Unit, 1013 Card Transmission Terminal, and 7701 and 7703 Magnetic Tape Transmission Terminals. In addition, the Communications Adapter affords a means of data interchange between the Model 20 and other models of the IBM System/360, including another Model 20.

1.2.2 Output Devices

- Output from the Model 20 CPU may take place through the IBM 1442 Card Punch Model 5, the IBM 2520 Card Read/Punch, the IBM 2560 Multi Function Card machine and either the IBM 2203 or 1403 Printer.
- Either the 1403 or 2203 may be installed but not both.
- Either the 2560 or 2520 may be installed but not both.
- Output to externally located data transmission units as well as to other models of the IBM System/360 can be accomplished by use of the Communication Adapter special feature.

The 2520, 2560 and Communications Adapter are described in the section on Input Devices.

IBM 1442 Card Punch Model 5

The IBM 1442 Model 5 provides punched card output for the System/360 Model 20. The Model 5 has a card hopper, a serial punch station, and a radial stacker. Card punching is done serially at a maximum rate of 160 columns per second.

Punching accuracy in the Model 5 is verified by comparing a signal, generated as each hole is punched, with data in the CPU core storage. Control of the 1442 is by the CPU stored program.

IBM 1403 Printer Models 2 and 7

The IBM 1403 Printer provides output for the System/360 Model 20 at a rate of 600 lines per minute. The Model 2 has a capacity of 132 printing positions, and the Model 7 can print 120 positions.

Single, double, and triple spacing of lines, plus skipping to a predetermined point are performed by the tape-controlled carriage, under control of the CPU stored program. The Model 2 has a dual-speed carriage that permits high-speed skipping.

Each printing position can print 48 different characters, and the printing format is controlled by the system's stored program.

As each character is printed, checking circuits are set up to ensure that the character printed is correct. Checks are also made to ensure that only valid characters are printed and that overprinting does not occur. If an error is detected, the machine stops, and the associated check light comes on.

IBM 2203 Printer Model A1

The 2203 Printer provides output for the Model 20 at up to 750 lines per minute. Interchangeable typebars allow the operator to select a type style and character set for a specific printing job.

The printing speed for any one application depends on the total number of lines printed; the amount of processing required for each printed line, and the character set used.

Single, double, and triple spacing of lines, plus skipping to a predetermined point, are performed by the tape-controlled carriage, directed by the CPU. The sequence and arrangement of data printed are also controlled by the stored program; a line to be printed is assembled in core storage is exactly the same sequence it is to appear as output. To ensure accuracy of output, each character is checked with the corresponding position in core storage before being printed.

The Dual-Feed Carriage special feature permits independent and simultaneous control of two set of forms.

1.3 MACHINE LANGUAGE

1.3.1 Character Code

- Data is stored as Extended Binary - Coded - Decimal Interchange Code (EBCDIC) bytes.
- Data is processed as half-bytes.
- Each half-byte is made up of a combination of five bits.
- One half byte is labeled Upper (U) and the other half byte is labeled Lower (L).
- Bits in the upper part of a byte are UP, U8, U4, U2, L1.
- Bits in the lower part of a byte are LP, L8, L4, L2, L1.
- P (Parity) bits are used to maintain an odd number of bits in each half-byte for checking purposes.

Description

All data is processed within the CPU as half-bytes except from or to Main Storage. Data from or to Main Storage is transferred as bytes. A byte is represented by an eight bit combination which gives 256 possible bit patterns (EBCDI Code). The bit positions of each half-byte consists of four value (8, 4, 2, and 1) bits, and one parity (P) bit. Numerical and alphabetic characters are represented in the 8 value bits of a byte as shown in the Code table (Figure 5-3). The P bit is used for parity checking purposes. Each half-byte within the computer must consist of an odd total number of bits or a parity error will be indicated. The P-bit will be present in a digit when the number of value bits present in a half byte is even.

1.3.2 Data Format

- Data is generally stored in Main-Storage to form fields and I/O fields.
- A field is a number of bytes related to internal processing.
- An I/O field is a number of bytes related to input-output operations.
- Fields are processed right to left; starting with the units position.
- I/O fields are processed left to right; starting with the high-order position.
- Data of a field can be in three different data formats:
  1. Fixed Point Data
  2. Decimal Data (Packed and Unpacked)
  3. Logical Data
- Data of I/O fields are Logical Data

Description

Data is stored in Main Storage to form fields or I/O fields. The length of these fields is different depending on the data format. A byte occupies only one Main Storage position and has its own specific address.

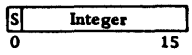
A field in Main Storage consists of a number of consecutive bytes and is composed of data related to arithmetic operations and internal field transmission. A field is addressed by its right-most (low-order) position which occupies the highest-numbered main storage position of the field. Fields are processed from the right to left into successively lower main storage positions until the process end-condition is detected. Process-end conditions depend upon which of the four different Instruction formats is being processed.

An I/O field in Main Storage consists of one or more fields of data related to input/output operations. An I/O field in Main Storage is addressed at the left-most (high-order) position which occupies the lowest numbered main storage position of the I/O field. I/O fields are processed serially from left to right into successively higher order main storage positions. I/O field transmission from or to input/output units is terminated when the I/O field, whose length is defined in the I/O instruction, is completely transferred.

Fixed-point Data

Fixed-point numbers have a binary radix and occupy fixed-length format consisting of a sign bit (s) followed by an integer field. The radix point of the integer is assumed to be immediately to the right of the low-order bit position. The integer field consists of fifteen bits and may be located in a general register or in a half-word of main storage.

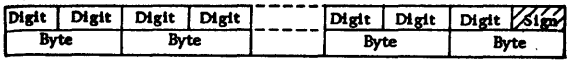
Negative numbers are carried in "twos-complement" form. The details of twos-complement arithmetic are described in the section on fixed-point arithmetic.



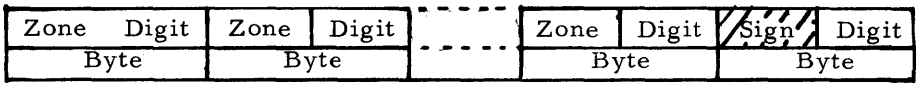
Decimal Data

Decimal numbers may appear in either the packed or the zoned format. In the packed format, an eight-bit byte contains two digits or, in the case of the low-order byte, one digit with a sign to the right. The packed format is used in all decimal arithmetic operations. In the zoned decimal format, one eight-bit byte contains only one digit, placed in the right-most four bit positions. The left four bits are considered zone bits, except for the low-order byte where they are used for the sign.

Packed Format. The Packed Format has a variable field length of from 1 to 16 bytes (1 to 31 Decimal positions).



Unpacked Format. The Unpacked Format has a variable length of from 1 to 16 bytes (1 to 16 Decimal positions).



The digits 0-9 are represented in a half byte (four bit binary coded decimal form) by 0000 to 1001. The codes 1010 to 1111 are not valid as digits. The sign is represented in a half byte by 1010 to 1111. Coding of plus and minus signs depends upon the code (EBCDIC or ASCII) in which the processor works.

Sign position bits	EBCDIC Mode	ASCII Mode
1010	+	+ Standard
1011	-	- Standard
1100	+ Standard	+
1101	- Standard	-
1110	+	+
1111	+	+

The zone in the unpacked format can be every bit configuration between 0000 and 1111.

The zoned format is not used in decimal arithmetic operations. Program instructions are provided for packing and unpacking decimal numbers so that they may be changed from the zoned to the packed format and vice versa. The processing of decimal numbers is described under Decimal Arithmetic.

Logical Data

A large portion of logical information consists of alphabetic or numerical character codes, called alphameric data. This logical information is used for communication with character-set sensitive I/O devices. The information has the variable field-length format and can consist of up to 256 bytes. It is processed left to right, one eight-bit byte at a time. Any of the 256 patterns of the EBCDIC or ASCII can be used to express Alphameric Data.

Logical information is handled as fixed-length and variable-length data. It is subject to such operations as comparison, translation, editing, bit testing, and bit setting.

1.3.3 Instruction Format

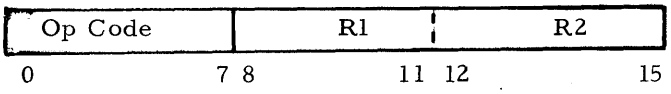
- Machine instructions are stored in General Storage.
- Four instruction formats, with a total of 37 instructions, are available to process the different data formats.

- The instruction formats are RR, RX, SI, and SS.
- Each instruction has a one byte operation code and normally has two addresses.
- The Op code specifies what is to be done.
- The two addresses (operand 1 and operand 2) specify the bytes to be processed.
- In the SI, RR, and RX format, two bytes are processed.
- In the SS format, the number of bytes that are processed is variable up to 256.
- The field length of bytes is designated in the byte following the Op code in the instruction.

RR and RX Format Instructions

To process fixed-point data, instructions in the RR or RX format must be used.

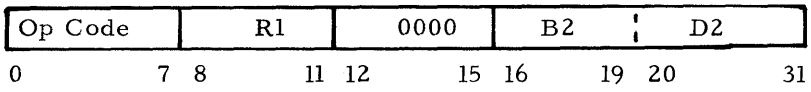
RR Format Instructions. The RR abbreviation refers to General Register to General Register operation (GR to GR). In the RR format, R1 specifies the address of a general register, the contents of which is the first operand.



In a fixed-point instruction which employs the RR format, the result replaces the first operand. The following are RR Type instructions:

Op Code	Name	Mnemonic	Type
0 0 0 0 0 1 1 1	Branch on Condition	BCR	Branching
0 0 0 0 1 1 0 1	Branch and Store	BASR	Branching
0 0 0 1 1 0 1 0	Add	AR	Fixed-point
0 0 0 1 1 0 1 1	Subtract	SR	Fixed-point

RX Format Instructions. The RX abbreviation refers to GR to Main Storage or Main Storage to GR operations. In the RX format, R1 specifies the address of a general register, the contents of which is the first operand. The address which designates the storage location of the second operand is derived from the contents of the B2 and D2 fields of the instruction



As described in the section on Addressing, the address of the second operand may be taken directly from the B2 and D2 fields or an effective address may be formed by adding the contents of the general register specified by the B2 field to the contents of the D2 field.

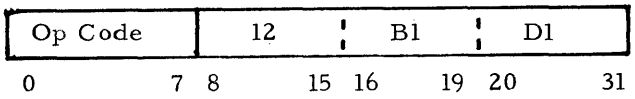
Results of operations replace the first operand except for Store (*Op code /40/). The result replaces the second operand for the Store operation. The following are RX type instructions:

Op Code	Name	Mnemonic	Type
0 1 0 0 0 0 0 0	Store Half word	STH	Fixed-point
0 1 0 0 0 1 1 1	Branch on Condition	BC	Branching
0 1 0 0 1 0 0 0	Load Halfword	LH	Fixed-point
0 1 0 0 1 0 0 1	Compare Halfword	CH	Fixed-point
0 1 0 0 1 0 1 0	Add Halfword	AH	Fixed-point
0 1 0 0 1 0 1 1	Subtract Halfword	SH	Fixed-point
0 1 0 0 1 1 0 1	Branch and Store	BAS	Branching

SS and SI Format Instructions

SI Format Instructions. The SI format instructions are used to process immediate data. Immediate data is part of the bit pattern in the instruction itself. The SI abbreviation refers to the transfer of immediate data from the instruction to Main Storage.

In the SI format, the address which specifies the core storage location of the first operand field is derived from the contents of the B1 and D1 fields of the instruction.



As described in the section on Addressing, the first operand address may be taken directly from the B1 and D1 fields or an effective address may be formed by adding the contents of the general register specified by the B1 field to the contents of the D1 field. The size of the first operand is one eight-bit byte.

The second operand is the eight-bit immediate data field (12) of the instruction.

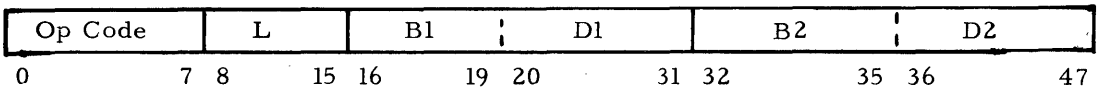
*The slashes (/) preceding and following an Op code indicate that the value is in a hexadecimal code.

The result of the operation replaces the first operand. The following are SI format instructions:

Op Code	Name	Mnemonic	Type
1000 0001	Set PSW	SPSW	Branching
1000 0011	Diagnostic		
1001 0001	Test under Mark	TM	Logical
1001 0010	Move	MVI	Logical
1001 0100	And	NI	Logical
1001 0101	Compare	CLI	Logical
1001 0110	Or	OI	Logical
1001 1001	Halt & Proceed	HPR	Logical
1001 1010	Test I/O and Branch	TIOB	Input/Output
1001 1011	Control I/O	CIO	Input/Output

SS Format Instructions to Process Logical Data. The SS Format instructions with one L (length) field are used to process locical data up to a process length of 256 bytes.

The SS abbreviation refers to Main Storage to Main Storage operations. The address which specifies the core storage location of the left-most byte of the first operand field is derived from the B1 and D1 fields of the instruction. The B2 and D2 fields similarly specify the left most byte of the second operand field. The first and second operand fields have the same length. The number of bytes extending to the right of the first byte is specified by the L field of the instruction.



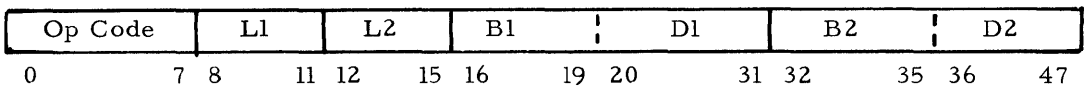
As explained in the section on Addressing, the address of each operand may be taken directly from the respective B and D fields of the instruction, or effective addresses may be formed by adding the contents of the general register specified by the B field to the contents of the D field.

The result of the operation replace the first operand field. The result is never stored outside the field specified by the address and length. The contents of the General Registers remain unchanged. The following are SS Format instructions to process logical data:

	Op Code	Name	Mnemonic	Type
D0	1101 0000	Transfer I/O	XIO	Input/Output
D1	1101 0001	Move Numerical	MVN	Logical
D2	1101 0010	Move Characters	MVC	Logical
D3	1101 0011	Move Zone	MVZ	Logical
D5	1101 0101	Compare	CLC	Logical
DC	1101 1100	Translate	TR	Logical
DE	1101 1110	Edit	ED	Logical

SS Format Instructions to Process Decimal Data. The SS Format instructions with two (L1 and L2) field lengths are used to process decimal data (arithmetic), up to a length of 31 decimal positions.

In this format, the address which specifies the core storage location of the left-most byte of the first operand field is derived from the contents of the B1 and D1 fields of the instruction. The number of operand bytes to the right of this byte is specified by the L1 field of the instruction. Therefore, the length in bytes of the first operand field is 1 to 16. This corresponds to a length code in L1 of 0000 to 1111. The second operand field is specified similarly by the L2, B2, and D2 instruction fields.



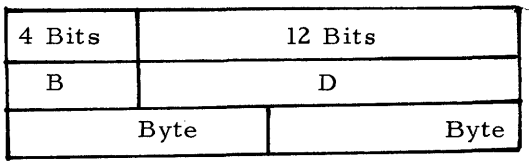
As described in the section on Addressing, the address of each operand may be taken directly from the respective B and D fields of the instruction. Alternately, effective addresses may be formed by adding the contents of the general register specified by the B field to the contents of the D field.

The results of the operation replace the first operand field. The result is never stored outside the field specified by the address and length. The second operand field remains unchanged, except in those cases where overlapping fields, which are permitted, actually occur. The following are SS Format instructions to process decimal data:

	Op Code	Name	Mnemonic	Type
F1	1 1 1 1 0 0 0 1	Move with Offset	MVO	Decimal
F2	1 1 1 1 0 0 1 0	Pack	PACK	Decimal
F3	1 1 1 1 0 0 1 1	Unpack	UNPK	Decimal
F8	1 1 1 1 1 0 0 0	Zero and Add	ZAP	Decimal
F9	1 1 1 1 1 0 0 1	Compare Decimal	CP	Decimal
FA	1 1 1 1 1 0 1 0	Add Decimal	AP	Decimal
FB	1 1 1 1 1 0 1 1	Subtract Decimal	SP	Decimal
FC	1 1 1 1 1 1 0 0	Multiply Decimal	MP	Decimal
FD	1 1 1 1 1 1 0 1	Divide Decimal	DP	Decimal

1.3.4 Addressing (Main Storage)

- An address used to refer to main storage may be specified by either of two methods; direct addressing or effective address generation (Indexing).
- Two bytes in an instruction are necessary to hold a main-storage address (direct or not direct address).
- The left-most 4 bits of the two bytes is the base (B) field, the remaining 12 bits is the Displacement (D) field.



- Direct addressing is indicated when the high-order bit in the B field is "zero".
- The remaining 15 bits are used to directly address main storage.
- Indexing is indicated when the high-order bit in the B field is one.
- The "one" bit in the high-order position of the B field gives, together with the remaining 3 bits of the B field, an address of any General Register (GR).
- The content of the addressed GR is added to the 12-bit of the D field in the main-storage address to generate an effective (Indexed) address.

Description

When the direct addressing method is employed, the low-order fifteen bits of the combined B and D fields of an instruction form an address which is used to refer directly to main storage, as follows:

Main-Storage Capacity	Direct Address	
	B	D
4K	0000	XXXXXXXXXXXX
8K	0001	XXXXXXXXXXXX
16K	0010	XXXXXXXXXXXX
	Greatest value in B Field	

Any one of eight general registers may be specified by the B field of an instruction as the location of the Base Address for effective Address generation.

B-Field	Register Number
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

NOTE: The eight general registers are numbered 8 through 15 for equivalence with their binary designations in the B-field. Registers 0 through 7 do not exist.

In the effective address generation method, the contents of the general register specified by the B field of an instruction are added to the contents of the D field of the instruction to form the effective address. The contents of the general register specified by the B field are referred to as the Base Address. The contents of the D field are referred to as the Displacement.

In forming an effective address, the Base Address is treated as an unsigned sixteen bit positive binary integer. The Displacement is similarly treated as a twelve bit positive integer. The two are added as sixteen bit binary numbers, ignoring overflow. A resultant sixteen bit sum which is greater than the main storage capacity installed in the CPU will result in an address outside available storage condition (Programing error).

The Displacement is a twelve bit number contained in the instruction format. The Displacement provides for relative addressing up to 4095 bytes beyond the base address.

The program may have zeros in the general register containing the Base Address or in the Displacement field of the instruction when effective address generation is specified. A zero indicates the absence of the corresponding address component. In this event, the effective address generated in the same as direct addressing with the non-zero component.

Initialization, modification, and testing of Base Addresses may be carried out by fixed-point, logical and branching instructions.

The first 144 bytes in main storage (addresses 0 to 143) are protected and are not available to the program. These bytes are employed for internal control by the CPU. An instruction or an operand address which refers to these bytes will result in an addressing error.

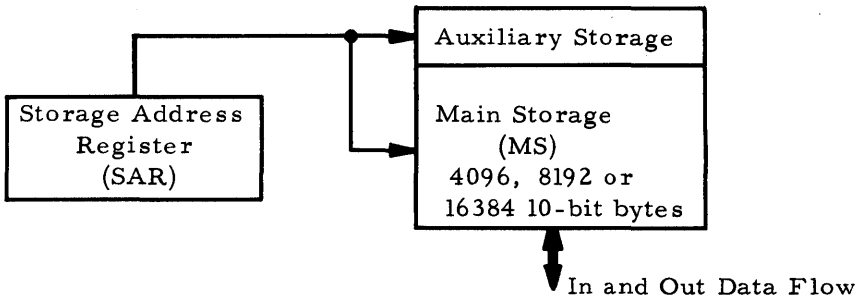
If one instruction contains two main-storage addresses, the B and D fields of the first operand are called B1 and D1. The B and D fields of the second operand are called B2 and D2.

I/O instructions have an operand 1 (B1 and D1), main storage address, and a B2 D2 address. The operand 1 address refers to the high-order address of the I/O field to be processed. The B2 and D2 is not used as a main storage address, but contains the field length (the number of bytes to be transferred by the I/O instruction).

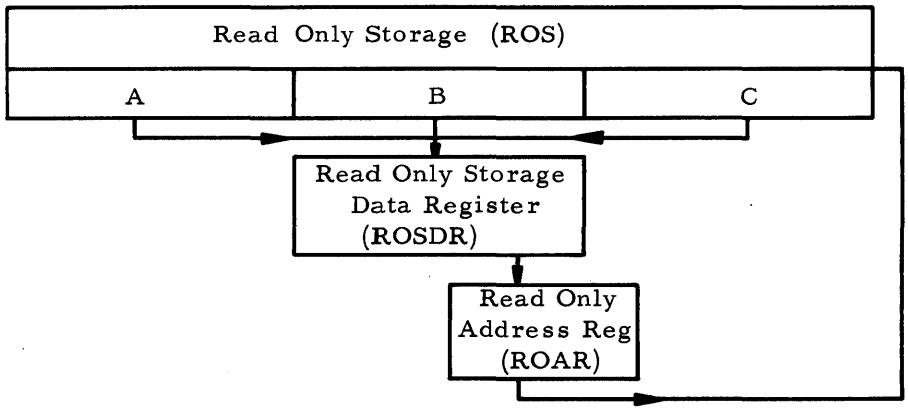
NOTE: All main storage addresses refer to the high-order position of a field. If it is necessary to process fields from right to left (low-order to high-order), the field-length number which is present in the instruction itself is added to the high-order address to get the low-order (units) address of a field.

1.4 SYSTEM CIRCUIT ELEMENTS

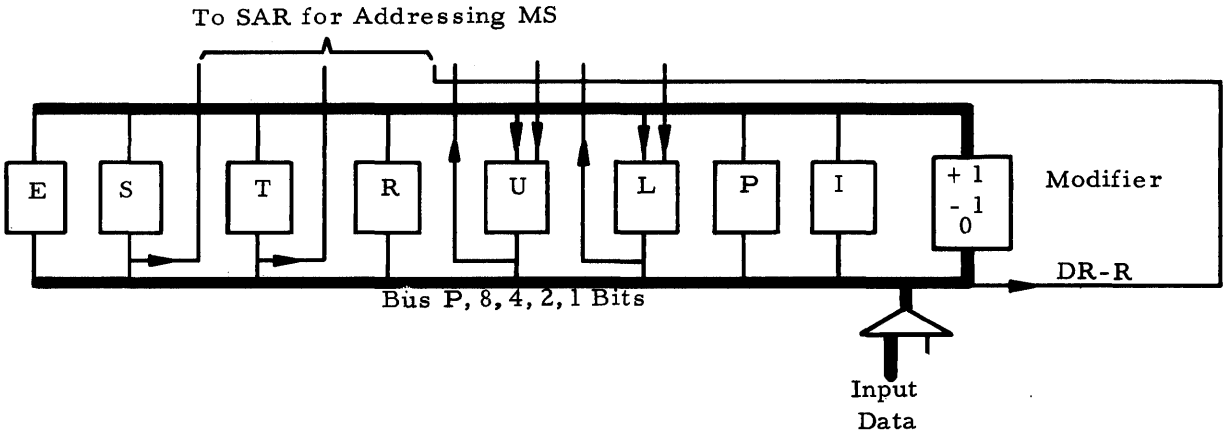
- The IBM 2020 CPU consists of 3 major circuit elements.
- A Main Storage with an Auxiliary Storage section and associated addressing provides the primary data storage



- A Read Only Storage with associated read out register and addressing circuits provides the basic instruction storage area.



- Eight Data Registers with an associated modifier circuit provide for all data manipulation.
- Eight Data Registers with an associated modifier circuit provide for all data manipulation.



- The 3 major circuit elements are connected to provide for processing of data (Figure 1-2).

Description

Programmed machine language instructions such as Add, Subtract, Multiply and so forth are processed in the IBM 2020 by means of a series of micro-programs which are permanently stored on plastic film tapes in the Read Only Storage (ROS) unit. Processing of a specific instruction such as Add involves reading out and sequentially performing a specific set of ROS micro-instructions.

The micro-program controls all movement of Data from the micro-instruction to the Data Registers, from one Data Register to another and from/to Data Register and Main Storage. The micro-program also controls the flow of input and output data snd the order in which instructions are performed. Provision is also made to branch from one micro-program routine to another.

Read Only Storage (ROS). The basic storage element in the ROS is a thin film tape on which copper conductor lines are etched. Holes are punched in the tape so that U shaped magnetic cores may be inserted to link selected conductors. Other holes are punched in the tape to break the copper conductor in selected places and thus prevent the linking of specific magnetic cores. Thus, the micro-program is defined by selective punching of the ROS tape.

Each ROS tape holds 3 micro-instructions, one 16-bit instruction (A in Figure 1-2) and two 22-bit instructions (B and C). Each micro-instruction includes a 10-bit configuration that defines one of the 15 different types of micro-instructions that may be performed. The remaining 12 bits give the location in ROS of the Next Sequential Instruction (NSI). In the 16 bit instruction, only 6 bits are used to give the location of the NSI, thus limiting the address of the NSI to a specific block of ROS.

Other circuit elements involved in read only storage circuitry are:

1. The Read Only Storage Data Register (ROSDR) holds the micro-instruction being performed. One of the three micro-instructions on each selected ROS tape is gated to the ROSDR. The NSI address is transferred to ROAR and the operation code output of ROSDR is analysed. Provision is made in the micro-program set to modify the next sequential address if required.

2. The Read Only Address Register (ROAR) holds the 22-bit address (modified by a micro-instruction if called for) of the next instruction to be performed. The ROAR output selects the next tape as well as the part of the tape A, B or C to be used as required.

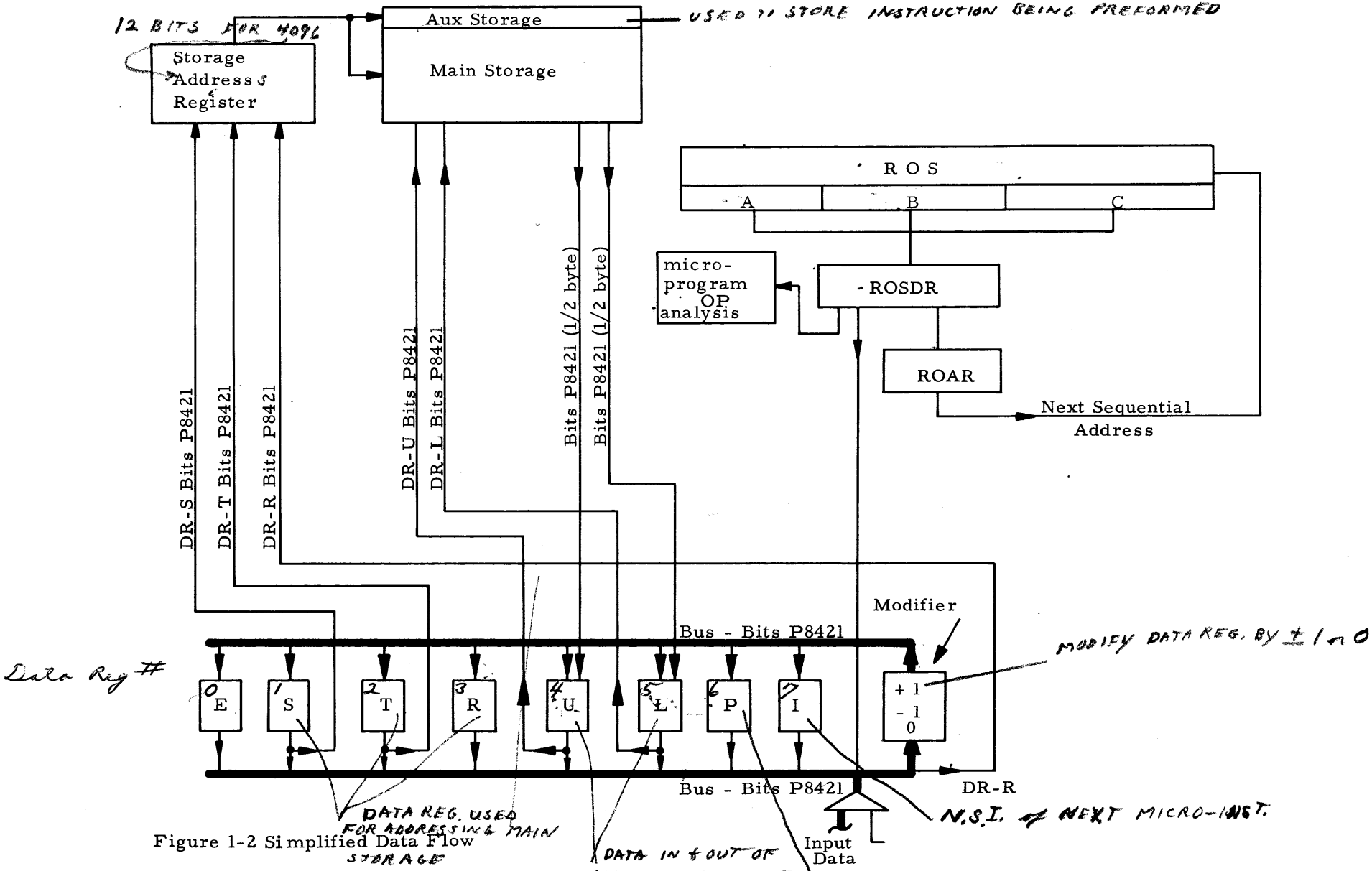


Figure 1-2 Simplified Data Flow

Data Registers (DR). Eight Data Registers, each holding the five bits P8421 provide for all data manipulation under control of the micro-program. Since each micro-program can define the movement of only five bits at a time, only one transfer of data from or to the DR can be made on a micro-instruction step. The Data Registers are used for the following basic purposes:

1. Addressing Main Storage is accomplished by DR-S, T, R. Since the micro-program can define the movement of only 4 bits (Plus the P, parity bit) at a time, multiple micro-instructions are required to move the required Main Storage address into DR-S, T, R and then to transfer the assembled address to the Storage Address Register (SAR). The SAR addresses a full byte (10 bits) of data for either re read in or read out.

2. Sending or Receiving data to or from Main Storage is accomplished by DR-U and L. On a read out from storage operation (Fetch), the 10-bit Main Storage byte addressed by SAR is read out into DR-U and L. On a read into storage operation (Store), the 10 bits in DR-U and L are transferred to the Main Storage location addressed by SAR.

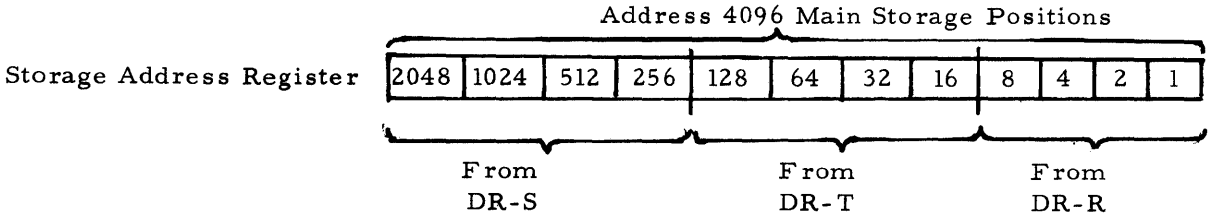
3. Output data paths and output controls are accomplished by use of DR-S, T, R, U and L.

4. Auxiliary Storage is addressed using the contents of DR-P. Auxiliary storage is used to store the machine program instruction being performed. It contains eight General Registers which are machine program addressable. Various address and data registers associated with the machine program, I/O buffer and I/O control registers are also included.

5. An indication of where the micro-program is to begin again after accomplishment of a sub-routine is set into DR-I.

6. All Data Registers connect to the Bus (Figure 1-2) and to the Modifier where they may be modified by ±1 or by 0. All DR may be involved in the general manipulation of data or addresses with in the CPU.

Main Storage. A Main Storage of 4,096 bits can be addressed by the 12 bits which are contained in DR-S, T and R and are subsequently transferred to the SAR.



If all positions of the SAR contain 1's, location 4096 is addressed. If all positions contain 0's, location 1 is addressed. Auxiliary latches contain two additional bits representing positions 4096 and 8192 to define storage locations for 8K and 16K memories.

1.5 SYSTEM OPERATION

- The System Reset key forces a specific micro-program starting address into the Read Only Address Register (Figure 1-3).
- The address forced into ROAR is the starting point of a micro-program routine.
- The Start key starts the micro-program.
- The micro-program routine tests such circuit elements as the Stop key and the Mode switch to determine what state the system is in.
- The IBM 2020 operates in alternate I-phase machine cycles (Instruction Phase) and E-phase machine cycles (Execution Phase).
- One I-phase and one E-phase are required to execute one machine programmed Op code.
- Multiple micro-instructions are performed during I and E phases.
- If the Mode switch is in the Run position, the micro-program continues with an I-phase during which the first machine program instruction is read out of Main Storage.
- During the I-phase, the machine program Op code is tested to determine into which of four formats it falls.
- The I-phase continues by loading Auxiliary Storage with the Op code, Operands and other data as required by the particular format of the operation.
- The Operation continues with one of the four types of E-phases micro-instruction sequences.
- During the E-phase, the micro-instructions associated with the programmed machine instruction are performed.
- The micro-program returns to the routine in which the Stop Key, Mode switch and so forth are tested before performing the next instruction.

Description

The outline of machine operation given above illustrates one method of starting a program. Many other factors are involved such as program load and preparing I/O equipment for operation. These are described in later chapters.

1.5.1 Micro-program Operations

- The 12 bit ROAR address selects a ROS tape.

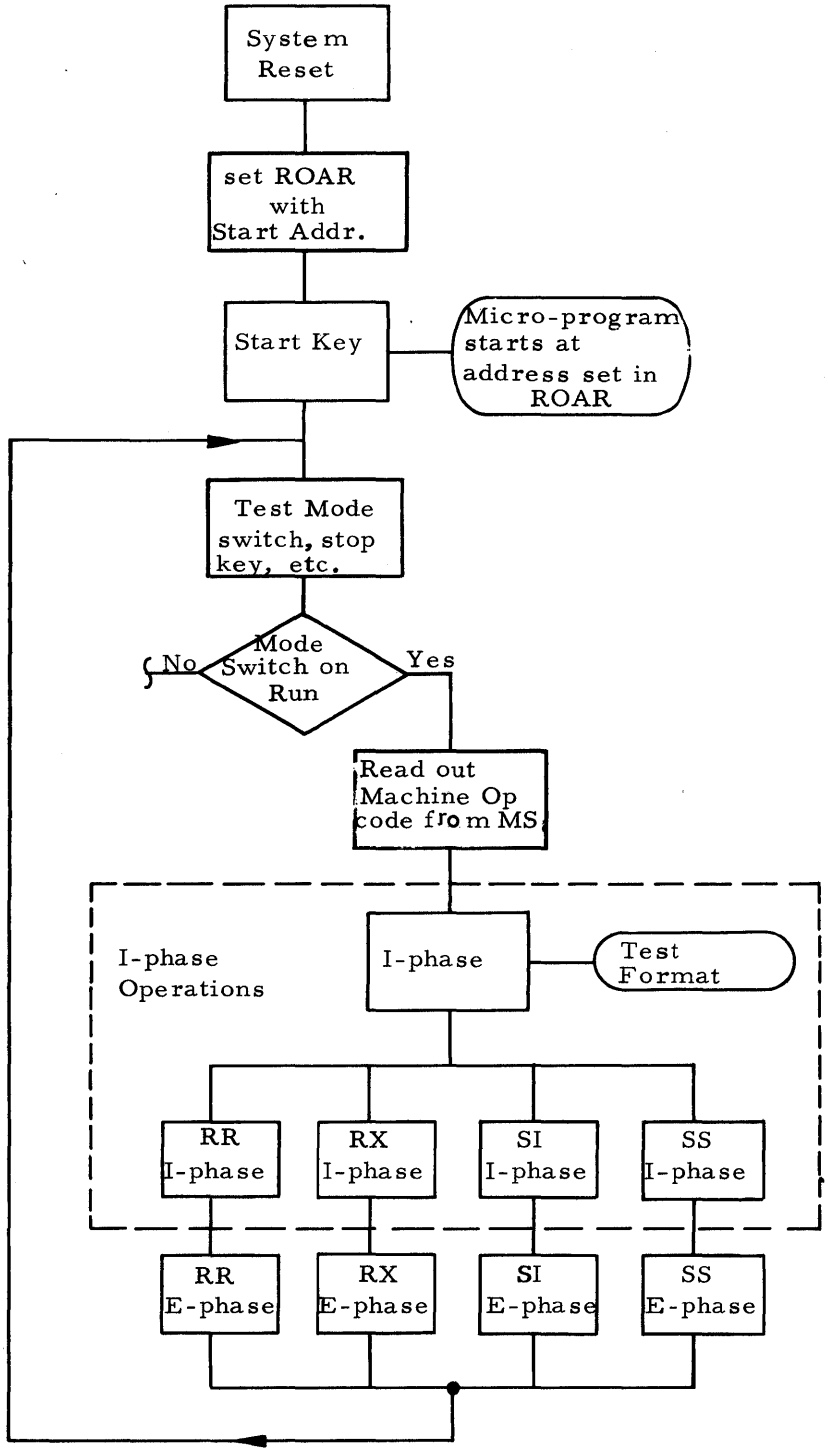


Figure 1-3 Simplified Machine Operation

- Sixty bits containing 3 micro-instructions are addressed.
- Only one portion of the addressed ROS tape (A, B or C-Figure 1-2) enters the ROSDR depending upon the ROAR address.
- Bits 0-9 of the ROSDR are decoded to specify which of 15 different micro-instructions is to be performed.
- Bits 10-21 are transferred from ROSDR to ROAR to supply the address of the next sequential instruction.

Description

The fifteen different instructions which may be stored in ROS are functionally divided into six groups as follows:

1. Six of the fifteen instructions are used to transfer data to or from Main Storage or Auxiliary Storage.
2. One instruction is used to transfer the content of one Data Register to another Data Register.
3. One instruction is used to transfer data from the operation part of the micro-instruction in the ROSDR to a Data Register.
4. Two instructions cause the content of any Data Register to be incremented or decremented by 1.
5. Two instructions, Sense and Control, allow the micro-program to sense the status of console switches and to communicate with the input/output device.
6. Three of the micro-instructions alter the four low order bits of ROAR and thus cause branching from one micro-instruction to another.

The six groups of micro-instructions are described in the following paragraphs:

Data Transfer from or to MS or Aux Storage. The six instructions in this group are divided into three types of operations. Two of the types (4 Instructions) are designed to read in (Store) and read out (Fetch) from Auxiliary Storage to DR U and L. One type of operation (2 Instructions) reads on or reads out from Main Storage locations.

1. Read in or read out any Auxiliary Storage location. Auxiliary Storage contains 256 ten bit bytes arranged in a 16 X 16 matrix (Figure 1-4). Each vertical column of the matrix is addressed by the contents of positions 6, 7, 8 and 9 of the operation code in the ROSDR. Each horizontal row is addressed by the contents of DR-P. A combination of the 4 bits in DR-P and the 4 bits in positions 6, 7, 8, 9 of the instruction in ROAR will address any of the 256 bytes in Aux Storage.

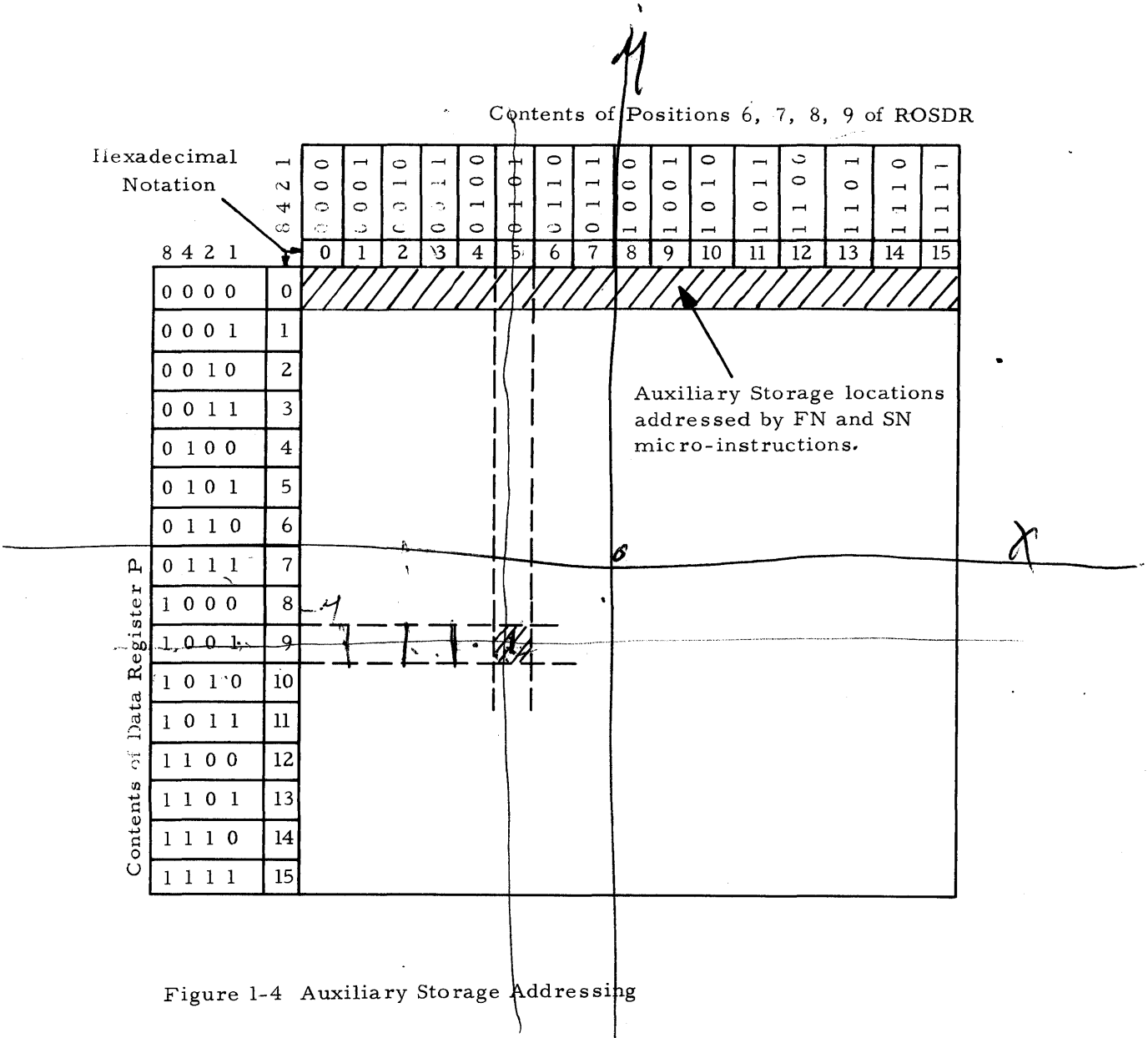


Figure 1-4 Auxiliary Storage Addressing

The functions in this category are defined as follows:

Function	Mnemonic	ROSDR Bit Structure 0 1 2 3 4 5 6 7 8 9
Fetch One Byte from Aux	F PN	P I 0 1 1 Y N N N N
Store One Byte in Aux	S PN	P 1 0 1 0 Y N N N N

In either instruction, the horizontal row of Aux Storage is addressed by the contents of DR-P. Data Register P must be loaded with the desired value before the F PN or S PN instruction is programmed. The value N N N N in the instruction addresses a vertical column of Aux Storage. For example, if 1 0 0 1 were placed in DR-P and the S PN or F PN instruction contained 0 1 0 1 in positions 6, 7, 8, 9 of ROSDR, the location shown in Figure 1-4 would be addressed.

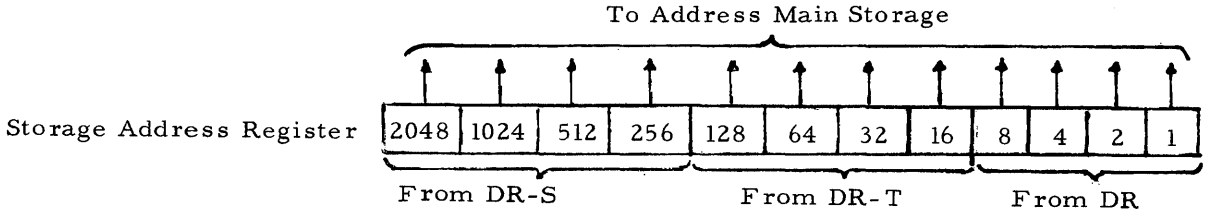
In the fetch instruction, the contents of the addressed Auxiliary Storage location are read out to DR U and L. In the store instruction, the contents of DR U and L are stored in the addressed Auxiliary Storage location.

2. Read in or read out horizontal row 0000 of Auxiliary Storage. The two instructions in this group operate in the same way as the instructions in group 1 except that DR-P does not have to be set. The value 0000 is forced into SAR to cause row 0000 to be addressed. The instructions in this category are defined as follows:

Function	Mnemonic	ROSDR Bit Structure 0 1 2 3 4 5 6 7 8 9
Fetch One Byte from Aux Row 0	F N	P 1 0 0 1 Y N N N N
Store One Byte in Aux Row 0	S N	P 1 0 0 0 Y N N N N

Positions 6, 7, 8, 9 of the instruction define the address in row 0000 of the byte to be read out. The FN and SN instruction allow fast access to much used data which is stored in Row 0000 of Auxiliary Storage.

3. Read in or read out any Main Storage location. The Main Storage address previously set into DR-S, T, R is used to locate the byte to be read into storage from DR-U, L or read out from storage into DR-U, L. The 12 bits set in DR S, T, R are transferred to SAR in order to address any of 4096 Main Storage bytes. If the storage contains 8,192 or 6,384 bits, auxiliary latches hold the required additional addressing bits.



The instructions in this category are defined as follows:

Function	Mnemonic	ROSDR Bit Structure 0 1 2 3 4 5 6 7 8 9
Read 1 byte from MS into DR U, L	F STR	P 0 0 1 1 Y 0 0 1 1
Write 1 byte into MS from DR U, L	S STR	P 0 0 1 0 Y 0 0 1 1

The ROSDR bit structure causes the transfer of the auxiliary latches and the content of DR-S, T, R into SAR. The bit structure also initiates a main storage cycle which causes either writing of a byte or reading of a byte.

Data Transfer from One Data Register to Another Data Register. Transferring data from one DR to another is frequently required in a micro-program. For example, when it is desired to store the last MS or Aux Storage address used, the address must first be moved from DR-S, T, R into DR-U, L so that it can be stored. This is done in 2 steps. First move DR-R into DR-L (Expressed as R ⟶ L) and then move DR-T into DR-U (T ⟶ U). The process is repeated to store DR-S and the content of the auxiliary addressing latches. A stored address may be moved back into DR-S, T, R by reversing the above procedure.

One instruction is used to move data from one DR to another:

Function	Mnemonic	ROSDR Bit Structure 0 1 2 3 4 5 6 7 8 9
Move DR-X to DR-X*	MX*X	P 0 1 X* X* X* 0 X X X

The contents of the DR expressed by bits 7, 8, 9 of the ROSDR are moved to the DR expressed by bits 3, 4, 5 of the ROSDR. The Data Registers are numbered 0-7 as shown below:

0	1	2	3	4	5	6	7
E	S	T	R	U	L	P	I

Thus, if the content of ROSDR is as follows, DR-T is moved into DR-U (T ⟶ U).

0	1	2	3	4	5	6	7	8	9
P	0	1	1	0	0	0	0	1	0

4= DR-U 2= DR-T

Data is moved out of the DR, to the bus, through the modifier (0) and to the receiving DR.

Move Data from the Operation Part of the Micro-instruction to a DR. Provision is made to transfer data or addresses from the micro-program instruction into any Data Register. The format of the instruction is as follows:

Function	Mnemonic	ROSDR Bit Structure
		0 1 2 3 4 5 6 7 8 9
Move NNNN into DR X*	MX*N	P 1 1 X* X* X* N N N N

A binary number (0 to 15) in the NNNN position of the ROSDR is moved into one of the DR (0-7) specified by the X* X* X* positions. Data is moved from positions 6, 7, 8, 9 of ROSDR to the bus, through the modifier (0) and to the addressed Data Register.

Increment or Decrement the Content of any Data Register by 1. The increment and decrement micro-instructions are used in micro-programs such as those designed to read a card. In such programs a Main Storage address into which the first card column is to be stored is specified. Also, the field length or number of card columns to be read is specified by the machine instruction. While the card is being read, the Main Storage address must be incremented by 1 and the field length must be decremented by 1. The increment - decrement micro-instructions accomplish this type of operation.

In the increment instruction, the contents of a specified Data Register are increased by one. In the decrement instruction the contents of the specified DR are decreased by one. The instructions in this category are defined as follows:

Function	Mnemonic	ROSDR Bit Structure
		0 1 2 3 4 5 6 7 8 9
Increment DR-X by one	Incr X	P 0 0 0 1 0 0 X X X
Decrement DR-X by one	Decr X	P 0 0 0 0 1 0 X X X

The bits in positions 7, 8, 9 specify which of the 8 Data Registers is to be incremented or decremented. The designated data register may contain any binary data 0 - 15. If the DR contains 1111 and an increment instruction is performed, a carry occurs and the DR goes to 0000. If the DR contains 0000 and a decrement instruction is performed, a carry occurs and the DR goes to 1111. In either case, the carry indicates that the particular micro-program subroutine which modified the DR has been completed. Provision is made in the circuit design for the carry to cause a branch to a new subroutine by modifying position 21 of ROAR to a 1. Position 21 must have previously contained a 0 in order for the branch to take place. The actual operation performed as a result of the carry depends upon the nature of micro-program subroutines.

The increment or decrement instruction is performed by reading out the DR addressed by positions 7, 8, 9 of ROSDR to the bus, through the modifier where it is modified by + or - one and then back to the addressed DR.

Sense and Control Input/Output Devices. In the Sense and Control instructions, the Console is considered to be an input/output device. The instruction operates in the same way whether console switches are being sensed or the condition of an external device is being tested. The sense and control instructions are similar but are described in two sections:

1. The Sense instruction is used in many ways, for example, it is necessary to sense the condition of the Mode switch to determine that it is in the Run position before leaving the manual routine micro-program to branch into the I-phase subroutine.

Each Sense instruction can sense four different conditions. Bits representing the conditions are set into DR-L by the Sense Instruction. The bits in DR-L are tested by the micro-program to determine the action to be taken. Sixty four different sense instructions may be programmed resulting in the ability to sense up to 256 different conditions.

Function	Mnemonic	ROSDR Bit Structure									
		0	1	2	3	4	5	6	7	8	9
Sense the condition specified by the n field. Set result in DR-L	Sense	P	0	1	n	n	n	1	n	n	n

The values in the ROSDR positions 3, 4, 5, 7, 8, 9 specify which of the 64 sense instructions (0 -63) is to be performed.

ROSDR Bits	3	4	5	6	7	8	9	
Binary Value	32	16	8		4	2	1	
	n	n	n	1	n	n	n	
Example	0	0	0	1	0	0	0	Perform Sense Inst 0, Sense Mode Switch

The Sense Table (Figure 5-7) shows the units tested by each of the 64 possible sense instructions. If the Mode switch is tested as shown in the example above and it is in the Char Displ position, DR-L would be set to 0 1 0 0 as a result of the sensing operation.

The sensed condition is gated onto the bus from the external device, through the modifier (0) and to DR-L.

2. The Control instruction operates in a manner similar to the sense instruction except that such operations as operating a card reader clutch or stopping the CPU can be controlled from the micro-program. Some of the 32 control instructions are conditional upon a certain bit in DR-U or L. Others are unconditional and will perform the designated function regardless of the content of DR-U or L. The use of the conditional control instruction expands the number of controls to more than 32.

Function	Mnemonic	ROSDR Bit Structure
		0 1 2 3 4 5 6 7 8 9
Perform the control designated by the n field, conditioned by the contents of DR-U, L as required	Ctl	P 0 0 1 n n 1 n n n

The values in the ROSDR positions 4, 5, 7, 8, 9 specify which of the 32 control instructions (0-63) is to be performed.

ROSDR Bits	4	5	6	7	8	9	
Binary Value	16	8		4	2	1	
	n	n	1	n	n	n	
Example 1.	0	0	1	0	0	0	Perform Ctl Inst 0, Reset Process Latch
Example 2.	0	1	1	0	1	0	Perform Ctl Inst 10, 2203 Printer Operation conditioned by DR-U, L

The Control Table (Figure 5-8) shows the units controlled by each of the 32 possible Control Instructions. Example 1 above is an unconditional control which resets the CPU Process latch. In example 2, if DR-U has been previously set with an 8-bit, a Set Car 1 Execute control is developed which causes the Carriage Execute latch to be set to start 2203 carriage motion.

Branching from One Micro-instruction to Another. Operation of micro-program sequences in performing machine instructions requires the ability to branch from one sequence to another. One example of a branching operation involves the sensing of external conditions such os Sense 0 which loads DR-L with a bit pattern. This bit pattern represents the condition of the Mode switch. The bit pattern may remain in DR-L or may be transferred to some other Data Register before the branch is accomplished.

In the example which follows, it is assumed that a Sense 0 instruction with a Next Sequential Instruction (NSI) address of XXX71 has loaded in DR-L. In location XXX71 is a branch instruction (also called Use instruction) which will test the contents of DR-L and cause the micro-program to branch to any of seven locations. (Figure 1-5). In these seven locations are subroutines which will process the operation indicated by the mode switch setting.

Instructions in the "Use" category are used to form a new micro-instruction address (NSA).

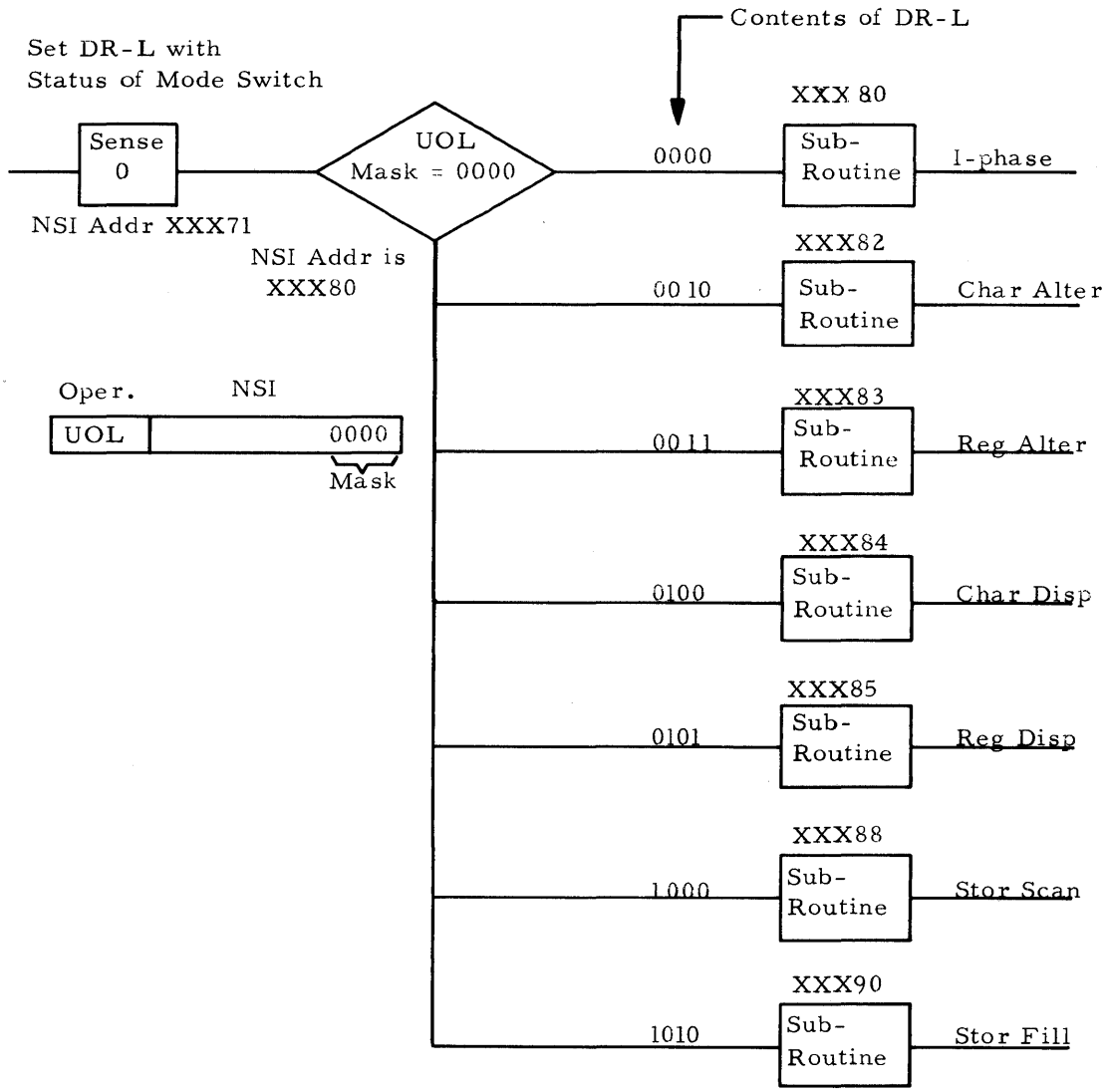


Figure 1-5 Use OR Instruction Example

Function	Mnemonic	ROSDR Bit Structure 0 1 2 3 4 5 6 7 8 9	NSI Address Mask 18 19 20 21
OR the content of DR-X with the NSI address (Mask) to create a branch address	UOX	P 0 0 0 1 1 1 X X X	M M M M
AND the content of DR-X with the NSI address (Mask) to create a branch address	UAX	P 0 0 0 1 0 1 X X X	M M M M
EXCLUSIVE OR the content of DR-X with the NSI address (Mask) to create a branch address	UXX	P 0 0 0 0 1 1 X X X	M M M M

In the instructions outlined above, the contents of the Data Register addressed by XXX is OR ed, ANDed or EXCLUSIVE ORed with the last four bits (Mask) of the Next Sequential Instruction. The result of the operation replaces the "Mask" bits in the NSI address in ROAR.

In the example (Figure 1-5), the bit structure of the UOL instruction is as follows:

Mnemonic	Instruction 0 1 2 3 4 5 6 7 8 9	Next Sequential Inst. 14 15 16 17 18 19 20 21
UOL	P 0 0 0 1 1 1 1 0 1	1 0 0 0 0 0 0 0
	DR-L	/8/ Mask

If the contents of DR-L are 0000 when the OR is performed, no change in the NSI is made, no branch occurs and the micro-program routine goes to the next I-phase and performs the instruction at address XXX80.

If the contents of DR-L are 0000, the OR function is performed as follows:

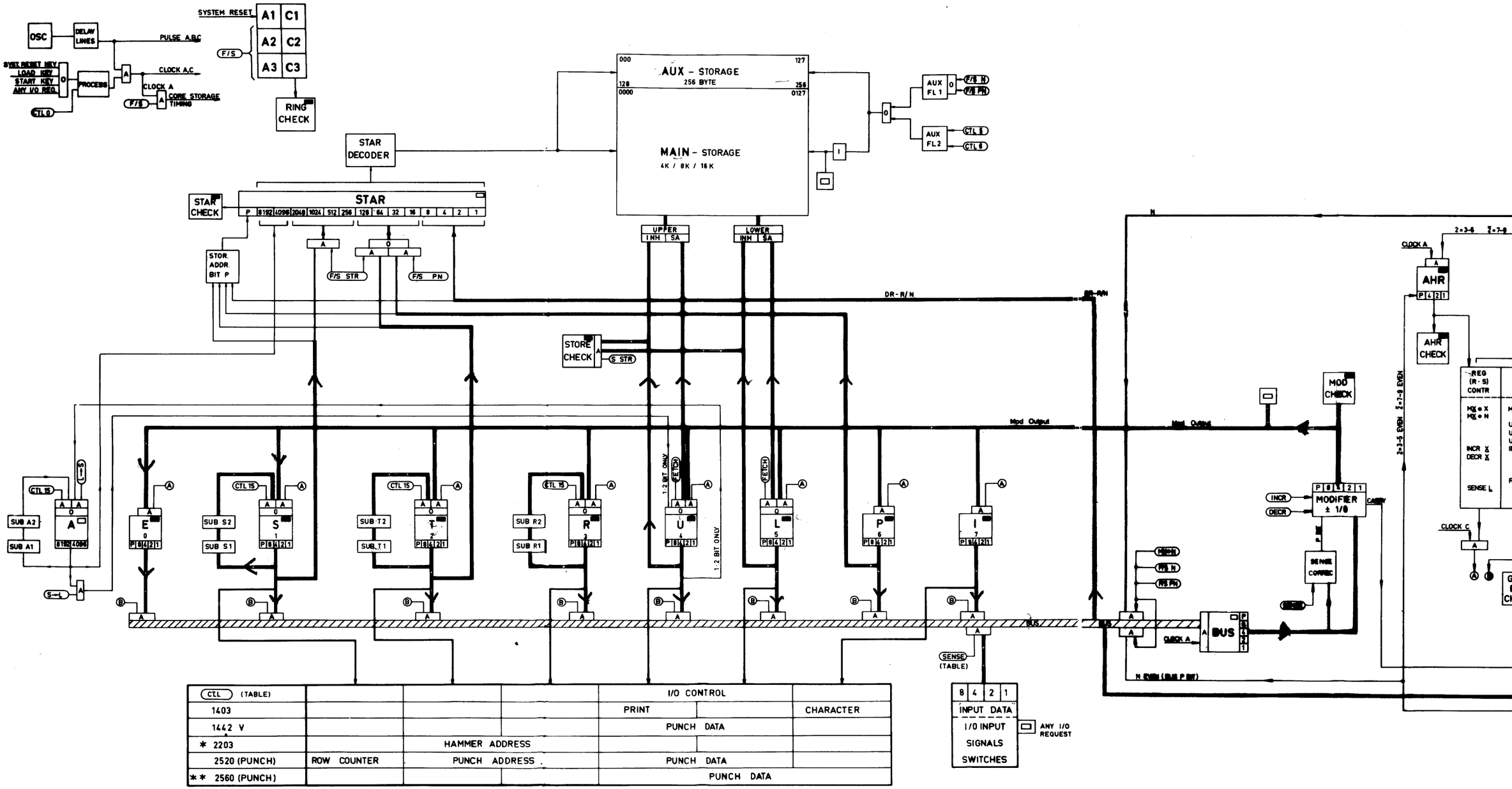
0000	Mask
0011	DR-L, If Mode switch was in Reg. Alter Position when sensed.
0011	PR Result = /3/

The result is stored back into ROAR and the next instruction is taken from location XXX83 where the micro-program routine to process the Register Alter operation begins.

The UAX and UXX instructions function in a similar manner except that a AND or Exclusive OR function is performed, for example:

AND Function	1 0 1 1	Selected Reg.
	1 1 0 0	Mask
	1 0 0 0	NSI Address placed in ROAR

Exclusive OR Function	1 0 1 1	Selected Reg.
	1 0 1 0	Mask
	0 0 0 1	NSI Address placed in ROAR

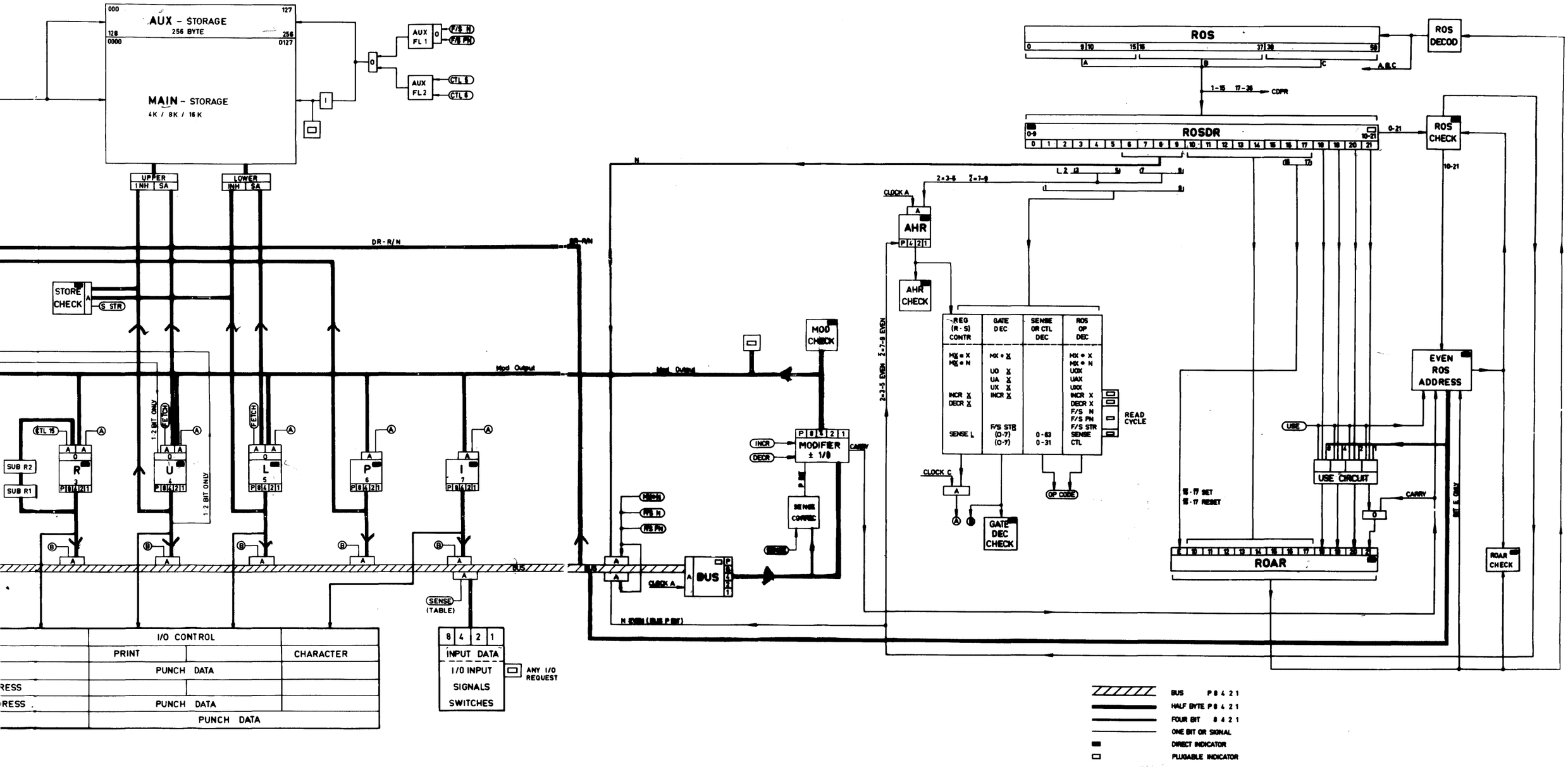


CTL (TABLE)				I/O CONTROL		
1403				PRINT		CHARACTER
1442 V				PUNCH DATA		
* 2203		HAMMER ADDRESS				
2520 (PUNCH)	ROW COUNTER	PUNCH ADDRESS		PUNCH DATA		
** 2560 (PUNCH)				PUNCH DATA		

* 2203 HAMMER FIRE AT MICROINSTRUCTION ADDRESS 3 F 1 E

* * 2560 (PRINT) PRINTPATTERN IN ROS WORD BIT POSITION 1-15, 17-36.

Figure 1-6 Data Flow


AUX - STORAGE
256 BYTE
000
127
128
256
0000
0127
MAIN - STORAGE
4K / 8K / 16K
AUX FL 1
AUX FL 2
F/S N
F/S PN
CTL 5
CTL 6
UPPER
INH
SA
LOWER
INH
SA
DR-R/N
DR-AN
STORE CHECK
S STR
Mod Output
MOD CHECK
CTL 15
SUB R2
SUB R1
R
U
L
P
I
1-2 BIT ONLY
FETCH
BUS
SENSE (TABLE)
INCR
DECR
MODIFIER ± 1/0
CARRY
SENSE CONTRL
SENSE
CLOCK A
N EVEN (BUS P BIT)
8 4 2 1
INPUT DATA
I/O INPUT SIGNALS SWITCHES
ANY I/O REQUEST
I/O CONTROL
PRINT
CHARACTER
PUNCH DATA
PUNCH DATA
PUNCH DATA
ROS
ROS DECOD
A B C
1-15 17-36
CDPR
ROSDR
0-21
ROS CHECK
10-21
AHR
AHR CHECK
CLOCK A
2=3-5 7=1-8
2=3-5 EVEN 7=1-8 EVEN
REG (R-S) CONTR
GATE DEC
SENSE OP CTL DEC
ROS OP DEC
MX ∘ X
MX ∘ N
UO
UA
UX
UOX
UAX
UBX
INCR X
DECR X
F/S N
F/S PN
F/S STR
SENSE CTL
F/S STB (0-7) (0-7)
0-63
0-31
READ CYCLE
OP CODE
SENSE L
CLOCK C
GATE DEC CHECK
EVEN ROS ADDRESS
USE
USE CIRCUIT
CARRY
15-17 SET
15-17 RESET
ROAR
ROAR CHECK
BIT E ONLY
BUS P 8 4 2 1
HALF BYTE P 8 4 2 1
FOUR BIT 8 4 2 1
ONE BIT OR SIGNAL
DIRECT INDICATOR
PLUGABLE INDICATOR

DATA REGISTERS

- The CPU 2020 contains 8 Data Registers (DR) which are micro program addressable.
- The DRs are labeled E, S, T, R, U, L, P, and I.
- Each DR consists of 5 latches (P, 8, 4, 2, 1) for containing a half byte.
- The major portion of data which are manipulated in the CPU pass through the DRs.
- All DRs can receive or transmit data from or to the bus.
- DR-S, T, and P are directly connected to STAR to address core storage.
- DR-U and L receive or transmit data to and from core storage.
- DR-S, T, R, U, and L are connected to the I/O devices for controls and output data paths.

Description

A DR can contain either data or addresses. Most DRs have specific functions in the data flow e.g.:

- DR-S, T, R normally contains the address which is used to address the main storage.
- DR-U contains the upper (U) part of the byte read out of or written into the core storage.
- DR-P normally contains the upper part of the address which is used to address AUX storage.
- DR-L contains the lower (L) part of the byte read out of or written into the core storage.
- DR-L is also used to store the sensed bits of a sense micro instruction.
- DR-E contains the contents of DR-A if the CPU is stopped for display purposes.
- DR-I often contains a constant, a binary number between 0 and 15, set into DR-I when the micro program enters a subroutine to indicate where the micro program is to continue after the subroutine is completed. I stands for Indicate.

SUB REGISTERS

- Data Registers A, S, T, R each have two sub-registers (Figure 2-10). Each sub-register holds one-half byte.
- Main storage is addressed by the content of the DR-S, T, R, (and DR-A).
- Sub registers are used to alternately address two fields in the main storage.
- One field address is located in the DR-S, T, R (and DR-A).
- The other field address is located in the associated Sub-register.
- An address exchange between the DRs and the SUB registers is executed with a CTL 15 micro-instruction.
- Figure 2-1 shows the controls and data paths of DR-R. The same control circuits are used for DR-S and T (and DR-A).

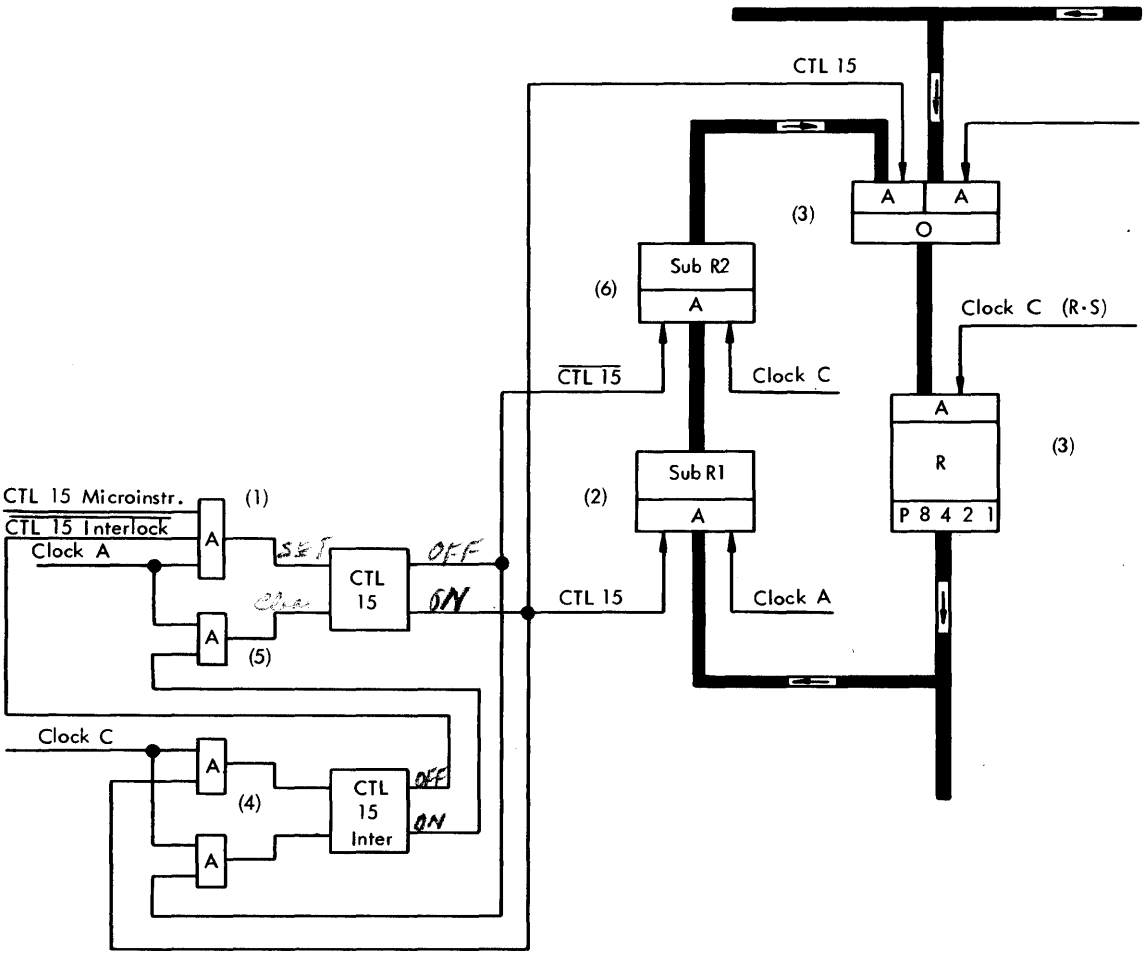
Description (Figure 2-1)

To execute an address exchange between a DR and a sub register, a CTL 15 micro-instruction must be given in the micro-program. One address is located in the DR-S, T, R, (and DR-A) the other address is located in the associated sub 2 register.

Circuit Description

Clock A (first cycle - See Figure 2-1) turns on the CTL 15 latch if a CTL 15 micro-instruction was decoded. The CTL 15 latch on side output is ANDed with the same Clock A to transfer the contents of DR-S, T, R (and DR-A) into the associated SUB 1 registers (-2-). CTL 15 latch On is ANDed with Clock C (first cycle) to transfer the contents, of the SUB 2.

* The numbers in parenthesis (-7-) refer to the circled numbers in the associated figure.



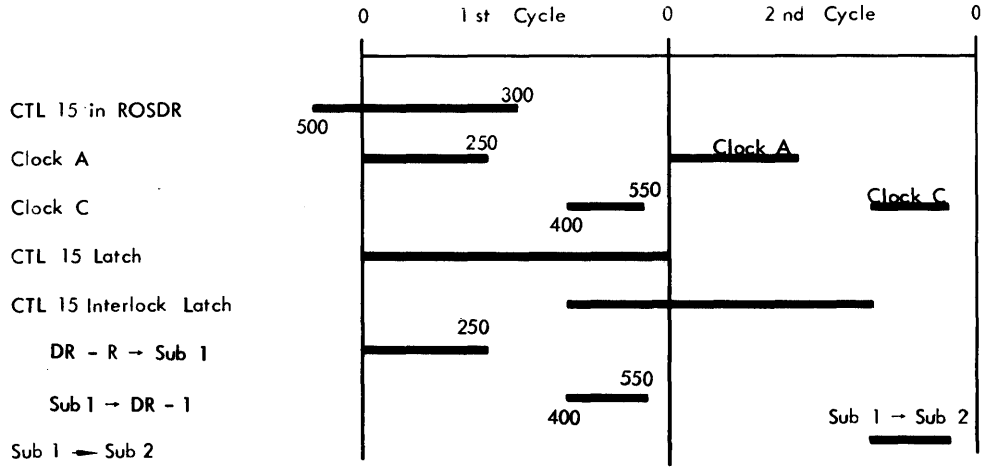


Figure 2-1. Sub-register and Sub-register Timing Chart

registers into the associated DR-A, S, T, R (-3-).

Clock C (first cycle) sets the CTL 15 Interlock latch if the CTL 15 latch is on (-4-). The next clock A is ANDed with the CTL 15 Interlock latch to reset the CTL 15 latch (-5-). CTL 15 latch off and Clock C (second cycle) transfers the content of the SUB 1 into the SUB 2 registers (-6-). When in the second cycle a CTL 15 micro-instruction is decoded again (two CTL 15 pulses in consecutive cycles) the CTL 15 latch is not turned on, because the CTL 15 Interlock latch is still on. This insures that no address is lost if two CTL 15 micro-instruction are given in consecutive cycles (improper micro-programming). The next clock C (second cycle) resets the CTL 15 Interlock latch.

BUS AND BUS LATCHES

- The Bus is a 5 bit channel: bits 8, 4, 2, 1, and P (parity).
- The Bus receives data from the DRs, I/O devices, and ROSDR position 6 to 9 (N part).
- The Bus transmits data to the STAR (pos. 1, 2, 4, and 8), ROAR (pos. 18 to 21), and into the Bus latches.
- Bus latches retain data during an entire cycle (Clock A to Clock A).
- Data on the Bus are not parity checked.
- Data in the Bus latches are parity checked at the output of the Modifier.

MODIFIER

- The Modifier receives data from the Bus latches.
- Binary data between 0 and 15 can be modified by +1, -1, or 0.
- Data passes through the modifier unchanged (modify by 0) when a micro-instruction other than INCR or DECR is used.
- The modifier changes data by one when an INCR or DECR micro-instruction is performed.
- A carry occurs during an INCR micro-instruction when the modifier input is 15.
- A carry occurs during a DECR micro-instruction when the modifier input is 0.
- The carry sets position 21 of the ROAR to cause a branch in the micro-program to another micro-program routine which will handle the carry.
- Modifier output is parity checked.
- In general, the output of the Modifier is transmitted into one of the DRs.

Description

See Figure 2-2.

TIMING (Figures 2-3 and 2-4)

- A 1.66 megacycle free running crystal oscillator produces 600 ns (0.6 μsec) cycles.
- A micro instruction is executed during one cycle.
- A cycle is divided into three signals: Pulse A, Pulse B, and Pulse C, by the "Cycle Timing Delay" circuit (-1-)*.
- The Pulses A, B, and C, time the start and stop circuits.
- Pulse A and C ANDed with the process latch generate Clock A and Clock C (-2-).
- Clock A enables the ROS Timing (delay) circuits (-3-).
- The output of the ROS Timing (delay) is ANDed with TROS Inhibit to generate the Driver Strobe and Reset Strobe signals.
- Clock A, and C, Driver Strobe, and Reset Strobe signals, time the micro-instructions and the data flow within the CPU (except for core storage timing).

*The numbers in parenthesis (-1-) refer to the circled numbers in the associated figure.

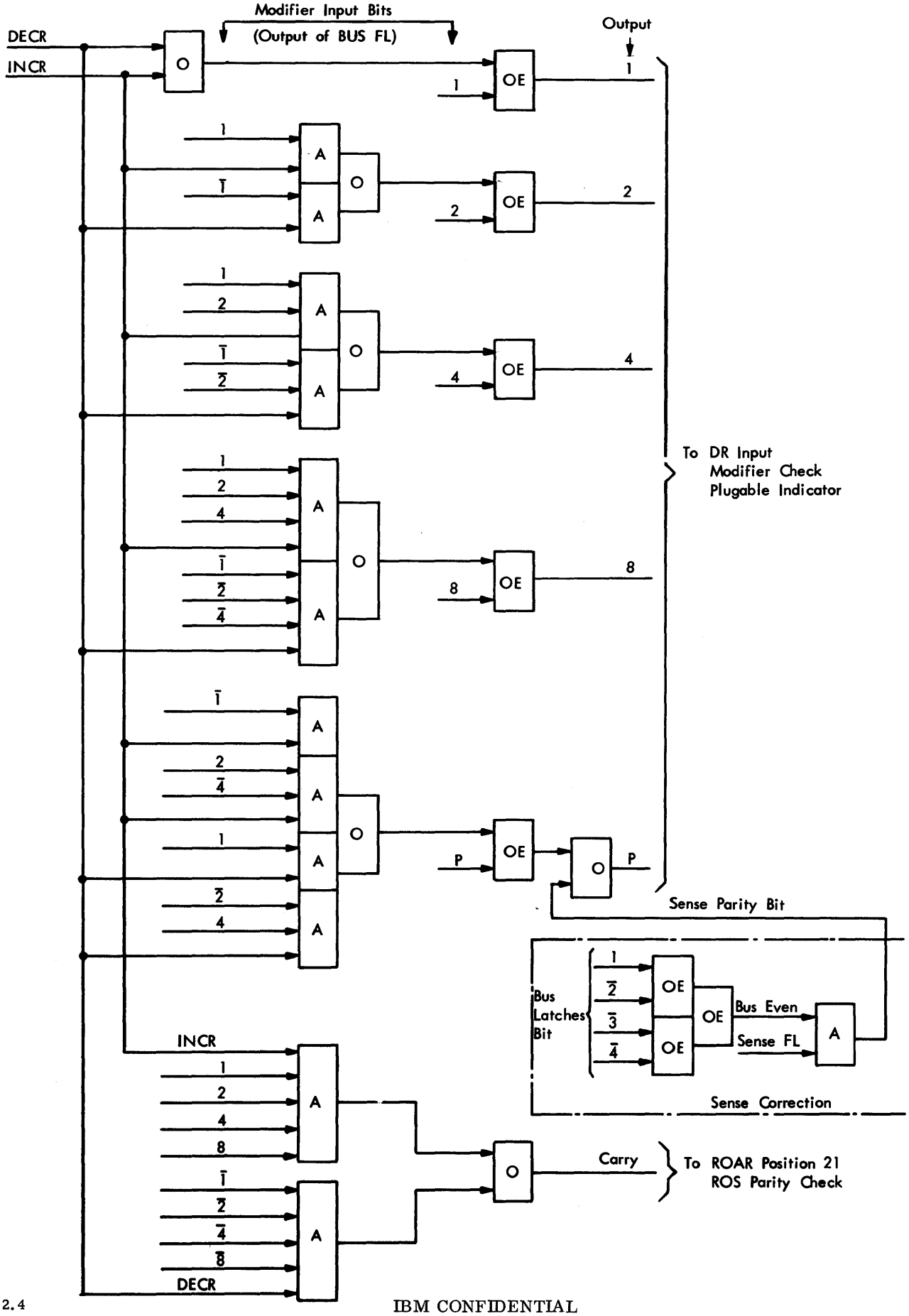
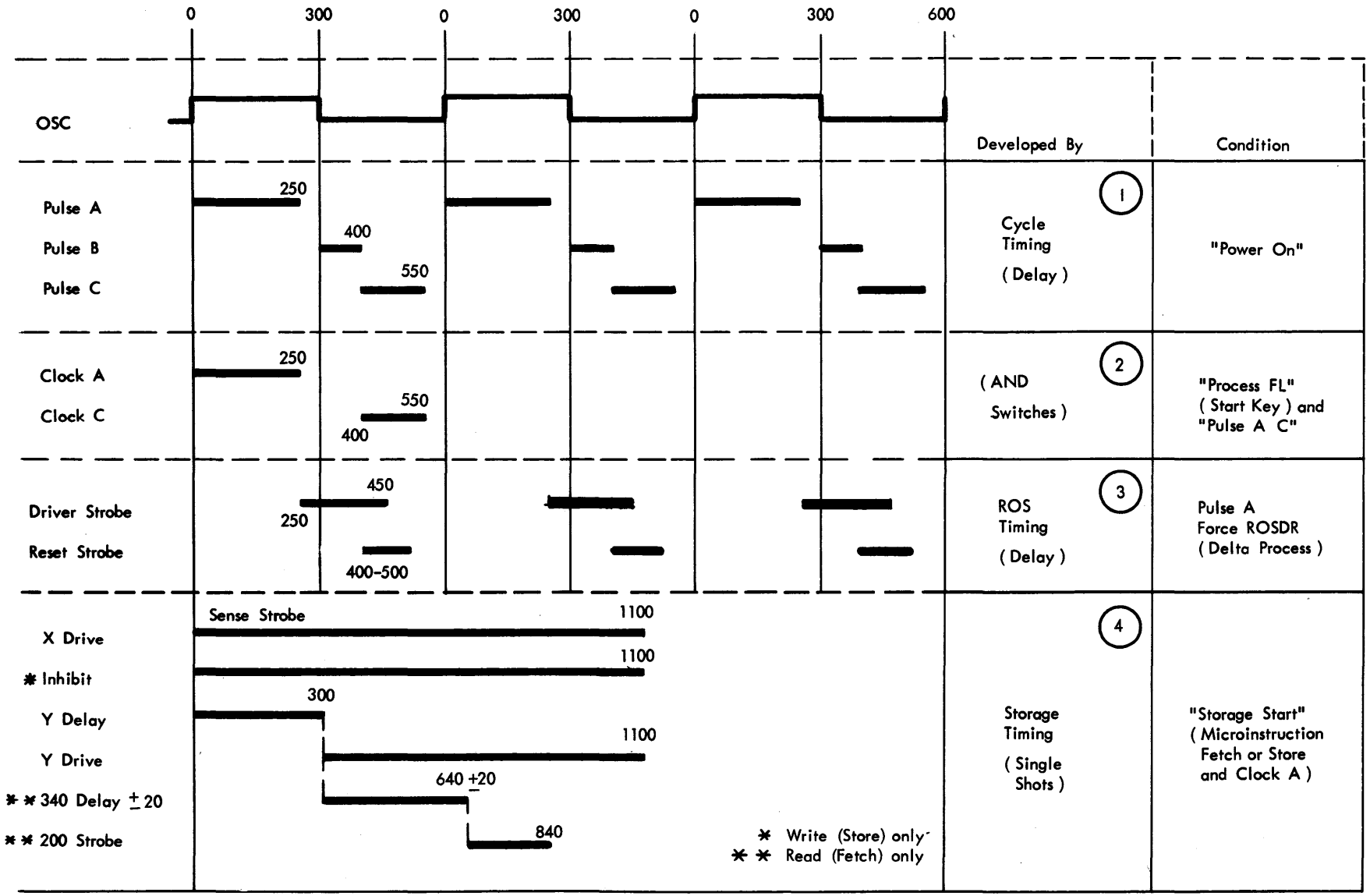


Figure 2-2. Modifier

Figure 2-3. CPU Timing Chart



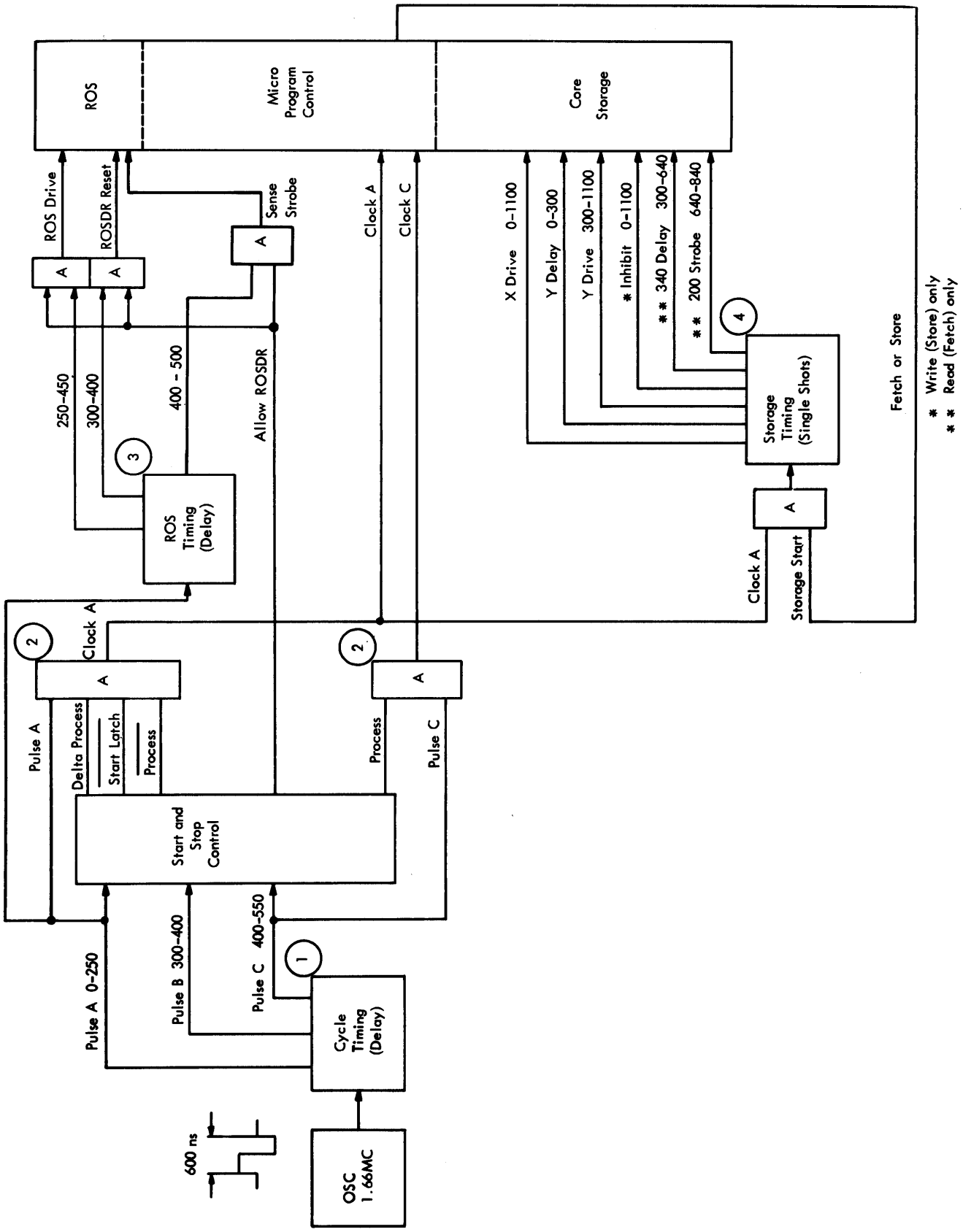


Figure 2-4. CPU Timing Diagram

- The core storage has a single shot timing circuit which generates all signals which are used to activate the storage for a read or write operation (-4-).

- The storage timing is enabled when a read or write (Fetch or Store) micro-instruction is decoded during a micro-program.

Description (Figures 2-3 and 2-4)

In the 2020 CPU, data is processed by micro-programs. To execute a single micro-program instruction, a timing device is necessary. One micro-instruction is read out of ROS and processed during a 600 ns cycle.

The 600 ns cycles are generated by a free-running crystal oscillator with a frequency of 1.66 MC (-1-). Each cycle is divided into three steps, Pulses A, B, and C by a "Cycle Timing (Delay)" circuit. The rest of the CPU timings are initiated with these signals.

Pulse A is ANDed with "Delta Process," "Not Start Key Latch," and "Not Process Check" to generate "Clock A." "Delta Process" is up when the Start key is pressed and no error is present. Not Start Key latch signal is up if the Delta Process latch is On and the next Pulse C is up (-2-).

Pulse C is ANDed with "Process" to generate "Clock C." (Process is up when the Start Key is pressed and no error is present.)

The TROS is in communication with the ROAR and the ROSDR. Clock A enables the TROS-Timing circuits to generate all signals necessary to control the TROS, the ROAR and the ROSDR. These signals are:

1. The Gate Strobe and Driver Strobe are used to select the proper Module and Tape in the TROS.
2. The Sense Strobe reads out the Word from the selected tape by ANDing the content of the ROAR positions 16 and 17 with the Sense Strobe. The signals: Sense Strobe A, B, and C are produced to distinguish between the 3 micro instructions which are contained in the one TROS Word.
3. Reset Strobe allows the ROSDR to receive a new micro-instruction from the ROS. The Reset Strobe is divided into Reset 1 and Reset 2. Both signals are normally present to reset the complete ROSDR. However, when the ROAR position 16 contains a zero bit (Indicating a short-instruction) the Reset 2 signal is dropped and the ROSDR positions 10-15 are not reset.

When a Fetch or Store micro-instruction is decoded in the micro-program, a Storage Start signal is generated which is ANDed with Clock A to start the Storage timing single shots.

The core storage requires a separate timing device, to generate storage signals such as X and Y Drive, Inhibit, and Strobe (-4-).

The "Storage-Timing Single shots" generate all signals necessary to activate the core storage for a read or write (Fetch or Store) operation.

Core Storage Timing

- When a Fetch/Store micro-instruction is decoded, the Start Read/Start Write signal is generated at Clock A time.

- Start Read/Start Write signal starts a Single-Shot circuit which generates timing signals that activate the Core-storage.

Description

1. One X line is activated from time 0 to 1100 depending upon the content (address) in STAR.
2. One Y line is activated from time 300 to 1100 depending upon the content (address) in STAR. The Y-line is delayed to reduce noise in the core storage.
3. The content of DR-U and L controls the Inhibit lines (10) which are activated from time 0 to 1100 during a Store operation.
4. During a Fetch operation, a Sense Strobe signal (time 640 to 840) is generated to gate the bit configuration of the byte which is read out into DR-U and L.

ROS Timing (Figure 2-5)

- Clock A (0-250) sets the address from the ROSDR into the ROAR.

- The output of the ROAR is decoded in the ROS Decoder and ANDed with the Driver Strobe (250-450) to read out the addressed word.

- The Reset Strobe (300-400) resets the ROSDR except for positions 10-15 when a short micro instruction is decoded.

- The Driver Strobe (250-450) and the Reset Strobe (300-400) signals are generated only if the Force ROS signal is active.

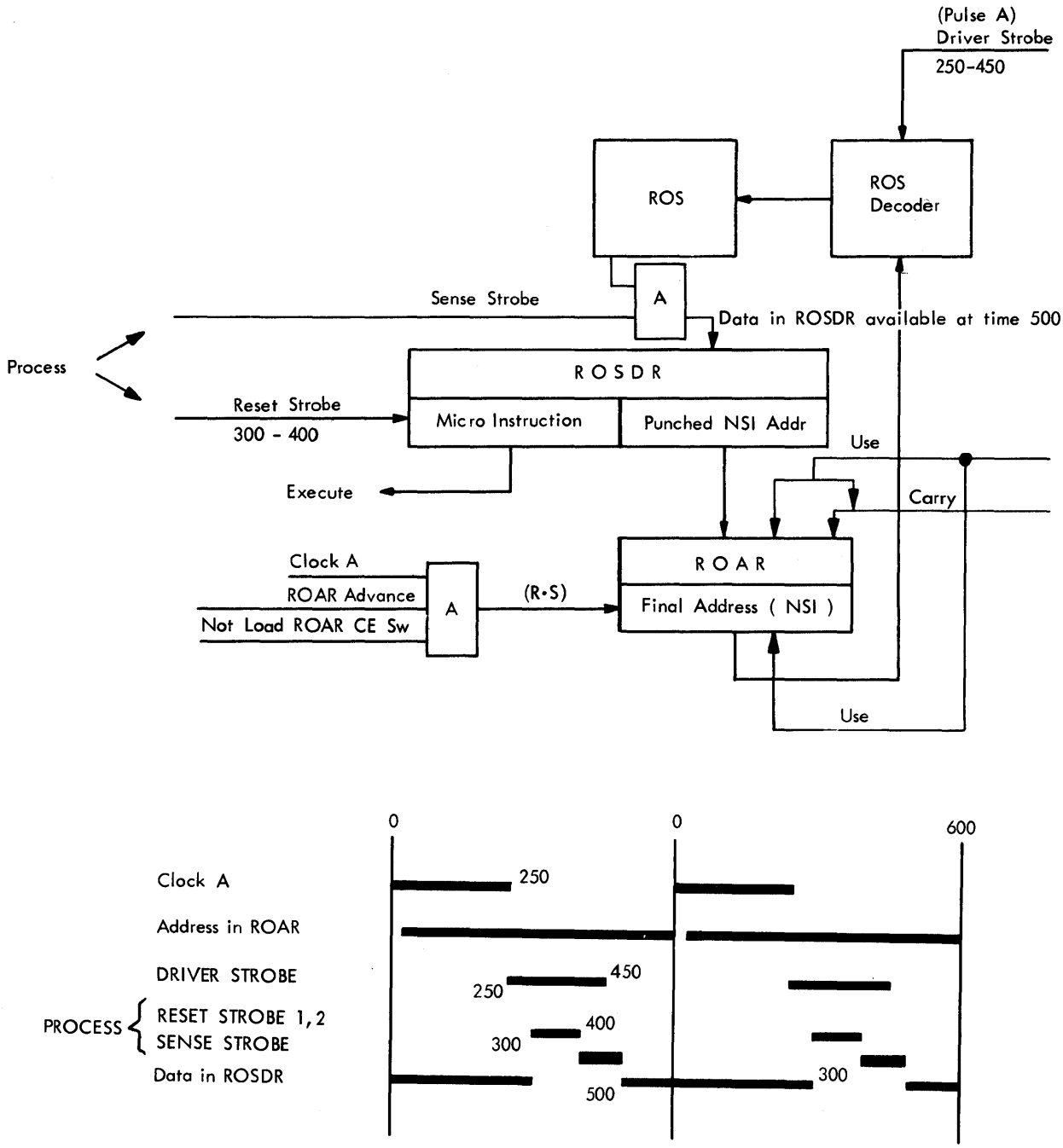


Figure 2-5. ROS Data Flow and Timing

MAGNETIC CORE STORAGE

- Three wire system with cores.
- Each physical storage plane in a 4K or 8K storage contains 8704 cores.
- In a 4K storage there are two bit planes in one physical core plane.
- In an 8K storage there is one bit plane in one physical core plane.
- One storage position consists of 10 bits.
- One 4K storage array holds 4096 Main Storage positions plus 256 auxiliary storage positions.
- One 8K storage array holds 8192 Main Storage positions plus 256 auxiliary storage positions.
- Addressing is accomplished by diode decoders connected to drivers and gates.

Magnetic Core Theory

- High retentivity of magnetic cores facilitates their use as storage devices.
- Direction of flux indicates stored binary value.
- Magnetization is accomplished by current carrying wire.
- Reversal of current causes a fast changing flux in a core.
- An adequate output voltage is induced in a wire when the flux direction changes.

Description

A magnetic core is a small doughnut shaped ring uniformly constructed of ferrite particles bonded together by a ceramic material. The ferrite particles have good magnetic properties and the core has a high retentivity of the magnetic flux lines after the magnetizing force is removed. It is this property of retentivity that makes a magnetic core useful as a storage device.

The operation of magnetic core can best be described by reference to the hysteresis curve (Figure 2-6). This curve is a plot of relationship between a magnetizing current (I_m) and the flux density.

A magnetic core is capable of maintaining indefinitely one of two stable magnetic states, either at point A or at point D on the hysteresis curve. Because the core has two stable states, it can be used as a binary storage device. At point A, the core has a residual flux in a negative direction, and at point D a residual flux in the positive direction. These directions can be arbitrarily assigned as binary "zero" and binary "one", respectively.

I_m is the amount of current necessary to change the state of the core. Plus I_m is an amount of current flowing in one direction, and minus I_m is the same amount of current flowing in the opposite direction.

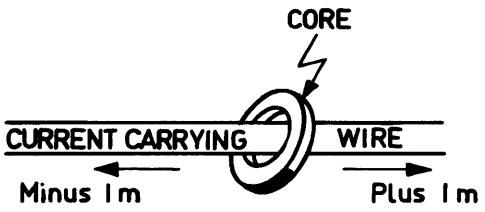
On the hysteresis curve, it can be observed that a magnetizing current of plus I_m will change the magnetism of the core from point A, a binary zero, to the magnetic saturation value in the positive direction at point C. When the current is removed, the total amount of magnetization drops back to point D (binary one). If, instead of full plus I_m, a current of plus $I_m/2$ were applied, the flux would change only the small amount from point A to point B on the curve, and when the current returned to zero, the flux would return to its original value at point A.

A reverse current, minus I_m, develops flux of opposite polarity and changes the magnetic field of the core from point D to the magnetic saturation value in the negative direction at point F. The total amount of magnetization drops back to point A (binary zero) when the driving current is removed.

Consider two parallel wires going through a core, each carrying half of the set current in the same direction. Set-current is the current required to change the magnetic state of a core. The resulting magnetic force is additive and equals the effect of full set-current. Now, shift one wire 90 degrees. With the same half-set current flowing, the effect is still additive, and produces enough magnetic flux to switch the core. These wires are called X- and Y- drive lines (Figure 2-7). With X- and Y-lines, a wire grid is constructed and cores are placed at the intersection of each X- and Y- line (Figures 2-8 and 2-9).

Principle of Coincident Current Addressing

- X- and Y- drive lines form a coordinate grid.
- In a 4K Storage, 34 X- and 128 Y- lines are used.
- In an 8K Storage, 66 X- and 128 Y- lines are used.



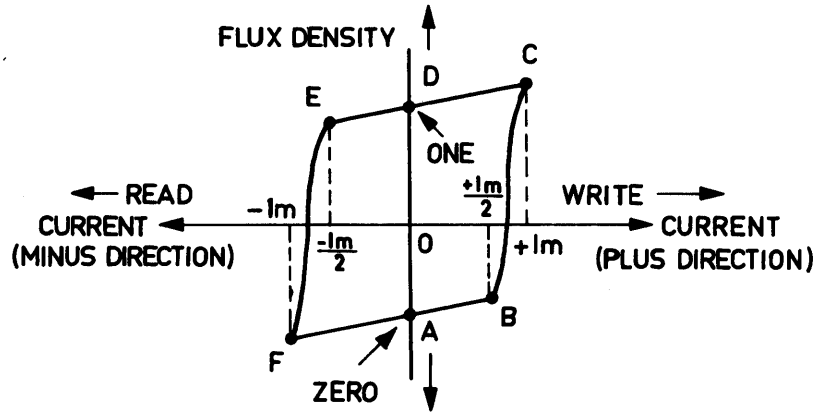


Figure 2-6 Hysteresis Curve

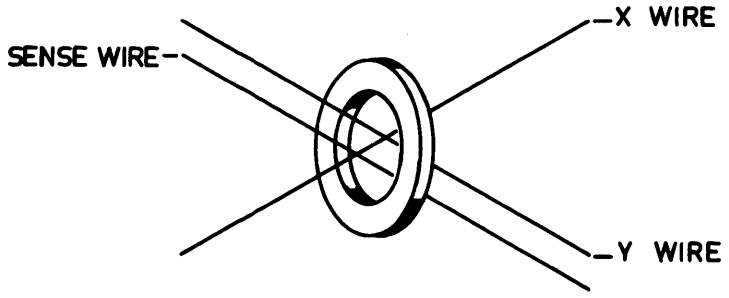


Figure 2-7. Ferrite Core

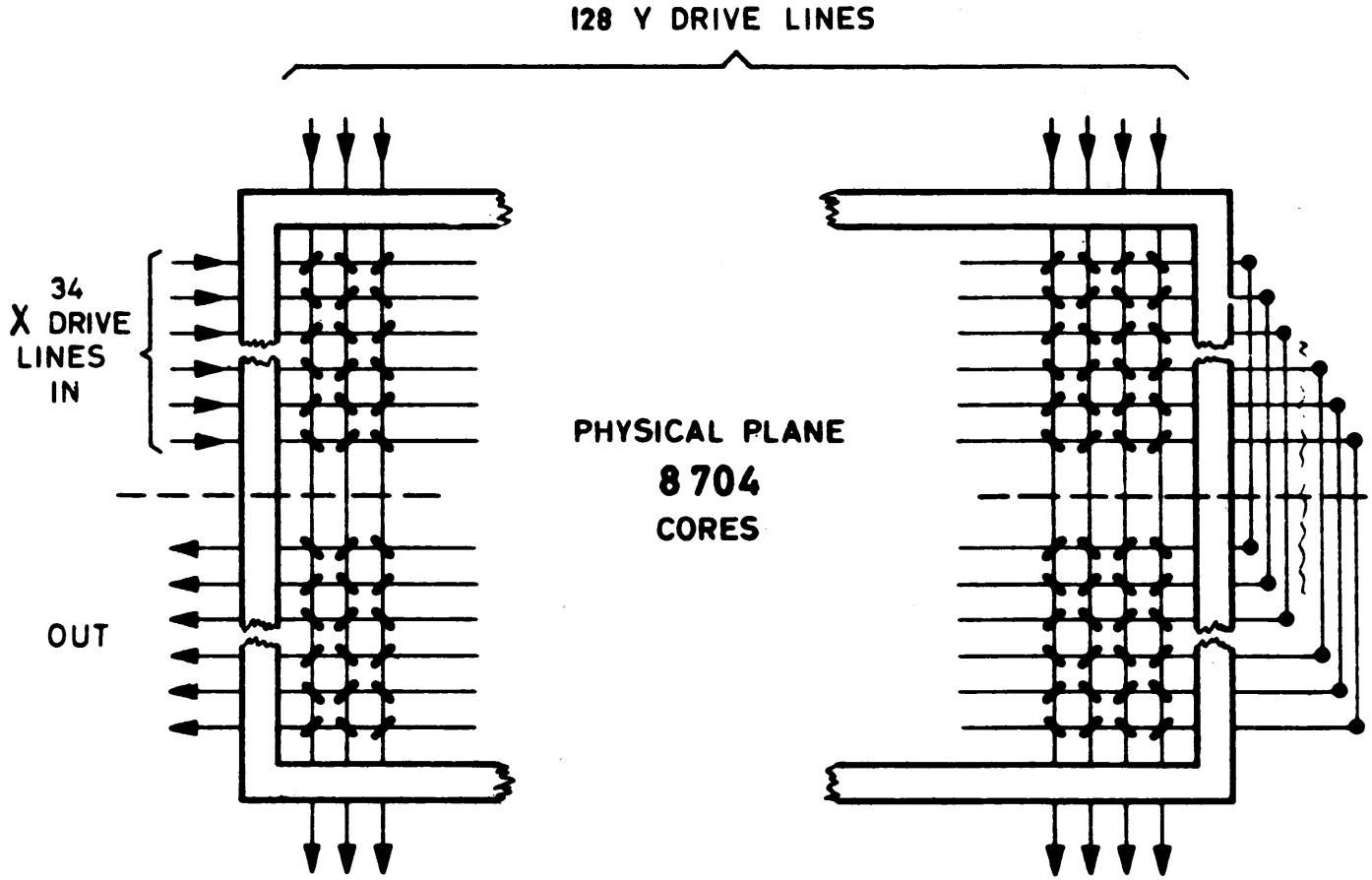


Figure 2-8 Layout of a Physical Plane 4K

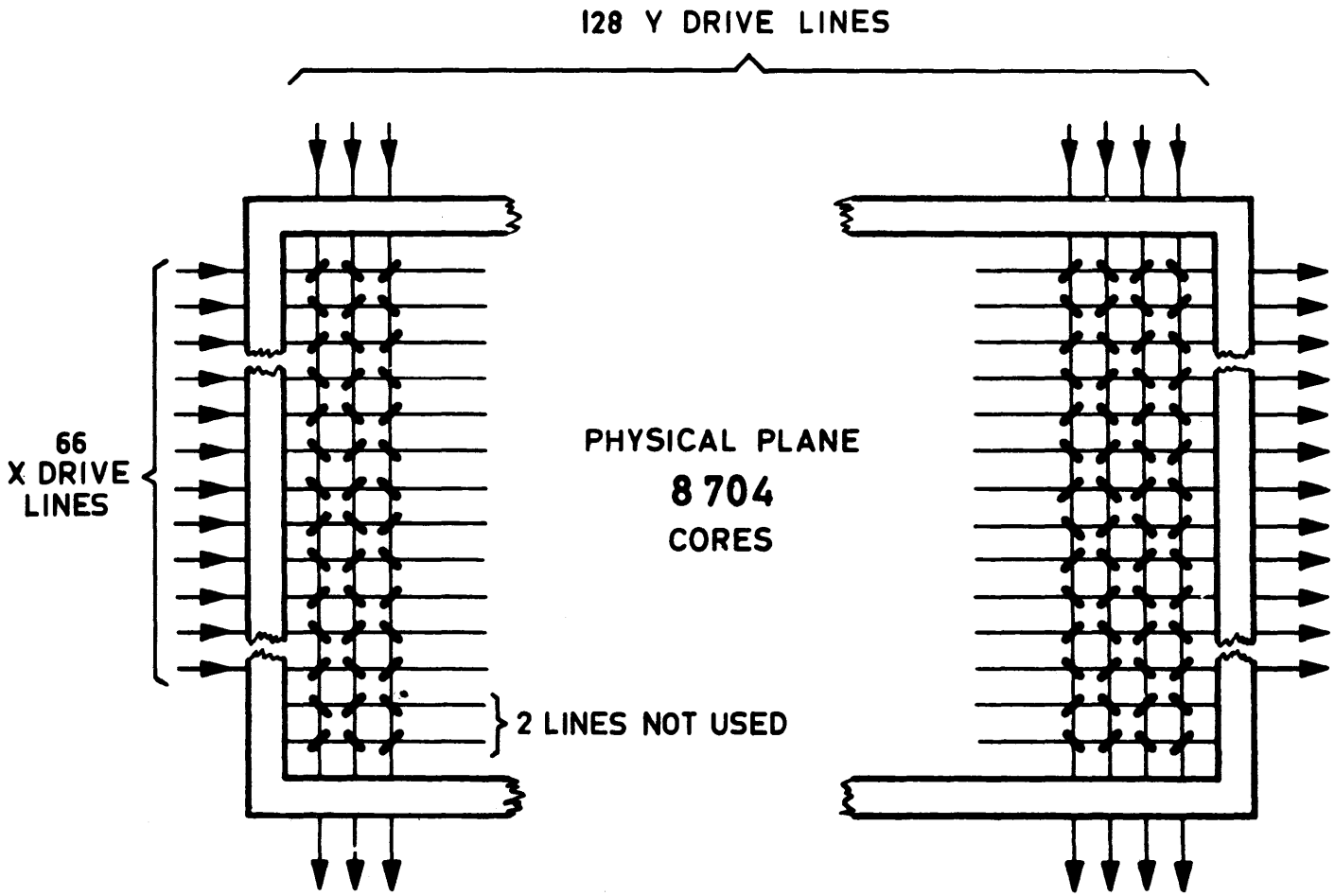


Figure 2-9 Layout of a Physical Plane 8K

Description

To accomplish addressing, half set-current is passed through an X- line and half set-current is passed through a Y line. Only the core at the intersection of the wires can be set, since only at the intersection is full set current produced. All other cores are not affected. By selecting the proper X- and Y- lines, each core in the grid can be addressed individually without affecting any other. The grid - assembly of wires and cores is called a plane. In a 4K array, one plane contains 4096 general and 256 special use cores. In an 8K storage, one bit plane contains 8192 general and 256 cores which are not used.

Figure 2-8 shows a plan for a 4K storage. Assume the jumpers on the right hand side are removed, thus considering only one half of the plane. With 34 X 128 (X and Y) drive lines, any one of 4352 cores can be addressed. One half of this physical plane constitutes one bit plane of the 4K storage.

Figure 2-9 shows a physical plane for an 8K storage. Here the physical plane is identical with one bit plane.

Core Array Assembly

- Ten bit planes per array.
- Five physical planes per 4K array (Figure 2-10).
- Ten physical planes per 8K array (Figure 2-11).
- One storage position contains ten bits; five bits in the upper (U) and five bits in the lower (L) part.

Description

Because information is stored in the Model 20 using ten bits per storage position, there is a need for 10 Bit planes. An array for a 4K storage is formed by adding the jumpers shown in Figure 2-8 thus having two bit planes per physical plane. Therefore five physical planes represent 10 Bit planes in the Model 20 Main Storage Array.

The ten bits are divided into upper and lower parts, each consisting of 5 bits; four data bits and one parity (P) bit. Note in Figure 2-10 that there is an U (Upper) bit part consisting of U1, U2, U4, U8, and UP and an L (Lower) bit part with L1, L2, L4, L8, and LP. Each part must maintain ODD parity (an odd bit count). This is accomplished with the UP bit for the Upper data bits and with the LP bit for the Lower data.

Core Array Wiring and Operations

- The core array has two separate half cycles; Read and Write.
- Each half cycle requires 1.8 μsec.
- It is not necessary that a Write half cycle immediately follow a Read half cycle.
- Sense and Inhibit functions are combined in one wire.

Write Cycle Operation

Figure 2-12 shows the 4K array with the X- and Y- lines for one storage position, that is for 1 byte (8 data bits and 2 parity bits). Since the X- and the Y- lines meet in each of the ten cores, energizing both lines in the direction shown results in setting the whole byte (10 cores) to the one state. Similarly, reversing the current in both lines results in setting all 10 cores to the zero state. The state (one or zero) is assigned arbitrarly. The Write Cycle is defined as an operation, during which the cores are set to the one state. Obviously there must be some control to set only specific cores to the one state. This control is attained with the combined inhibit sense line. This third wire is strung through all cores of a bit plane. Each bit plane has it's own inhibit/sense line. (Figure 2-13). The current in the inhibit/sense line is opposite to the current in the X- and Y- line. Thus, when the inhibit line is energized during a Write cycle, the sum of the X- and Y- current is effectively cut to half. With only half of its set current flowing, a core cannot be set. Thus, by energizing the proper inhibit lines when addressing a byte, the proper bit pattern can be set.

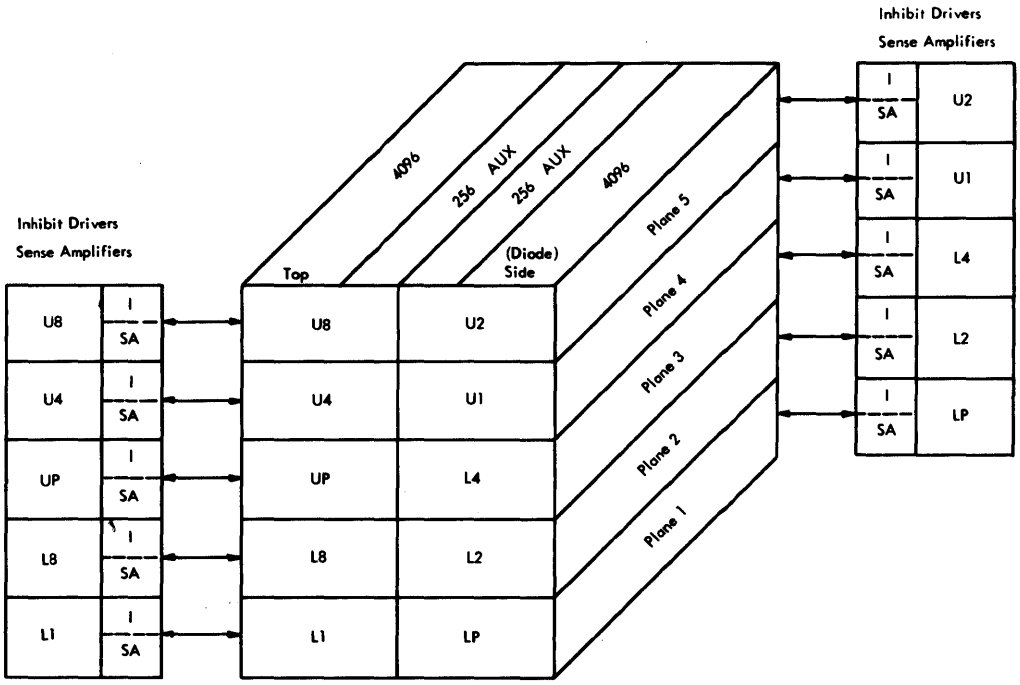


Figure 2-10 Bit Plane Arrangement 4K

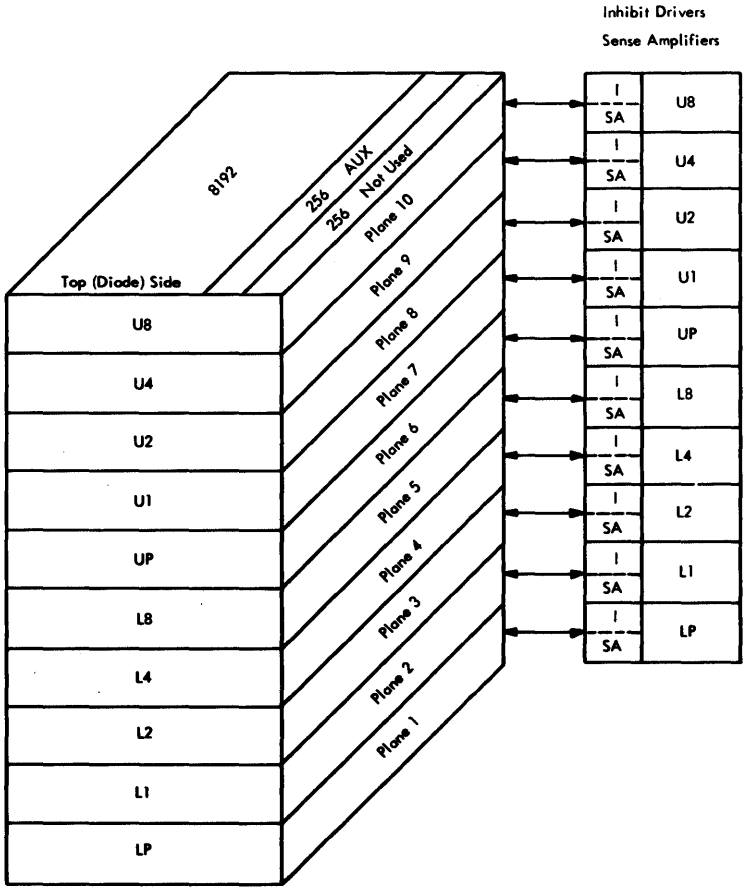


Figure 2-11 Bit Plane Arrangement 8K

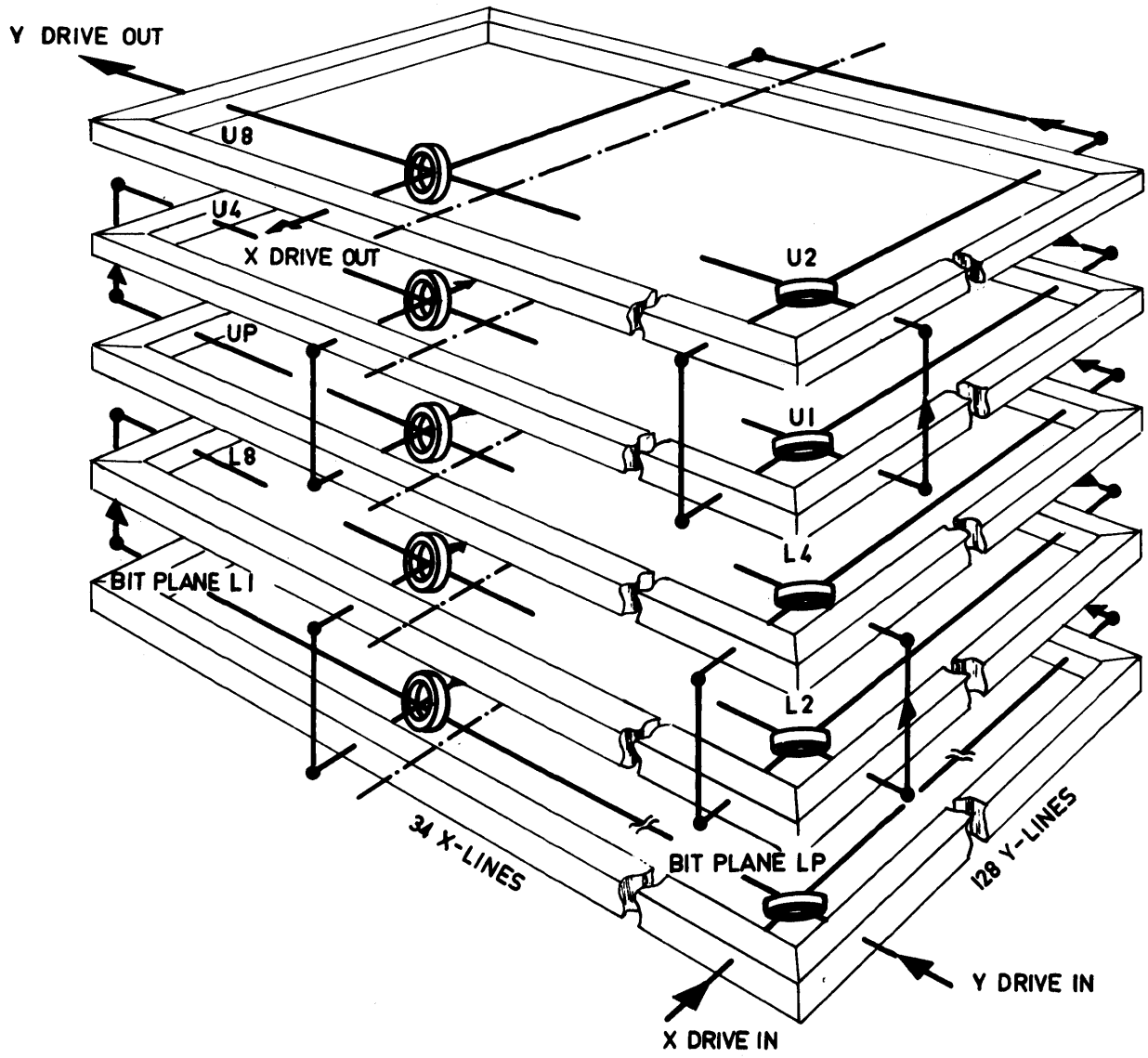


Figure 2-12 Path of X and Y Drive Lines

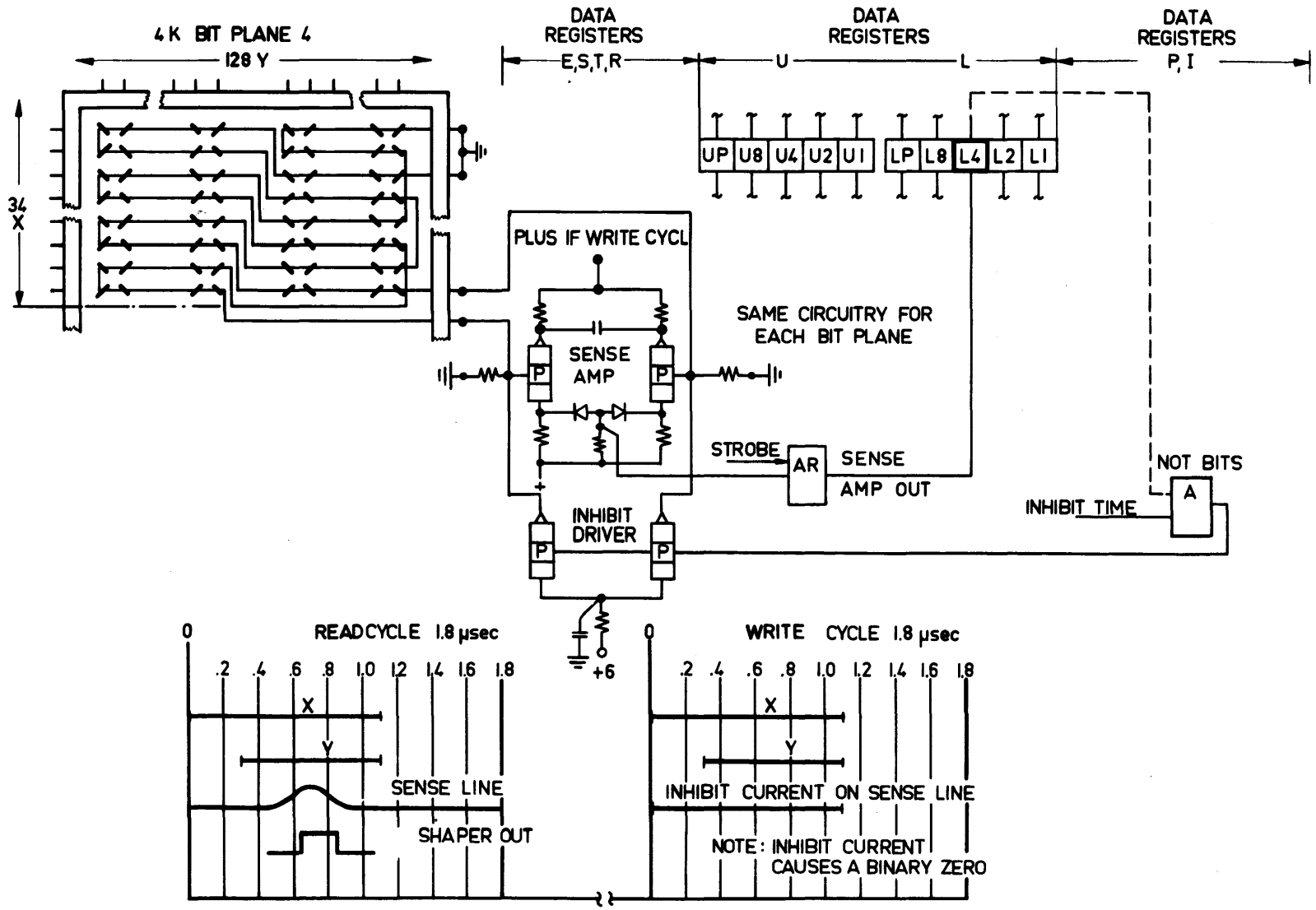


Figure 2-13 Sense and Inhibit Circuitry with Timing

Read Cycle Operation

During the Read Cycle, the X- and Y- line is energized, this time in a direction opposite to that in the Write Cycle. This means setting all those cores to zero that were previously in the one state. When these cores flip to zero, the abrupt change in their magnetic fields induces a current in their respective inhibit/sense lines. During the Read Cycle, the inhibit/sense lines are not energized as they now function strictly as sense lines. In this manner all cores that contained a one bit are read out (the others are not affected). Driving cores to zero that were already zero, produces a very small signal in the sense lines. The sense amplifiers amplify the strong signals from flipping cores only. Reading out in this manner sets all cores to zero. This type of operation is therefore called destructive readout. The information that was in the cores is now preserved in DR-U and L since the sense amplifier outputs set the appropriate latches in these registers. The preserved information can be written back or the contents of DR-U and L can be altered and the new contents can be written back into the same storage position.

ADDRESSING SYSTEM

- To address an 8K Main storage, the binary address is contained in DR-S, T, R and the high order position in DR-A (Figure 2-14).
- To address an 8K Main storage, the binary address is contained in DR-S, T, R and the high order position in DR-A (Figure 2-14).
- STAR is set by the contents of DR-S, T, and R, and DR-A (With an 8K storage).
- The binary address in STAR activates the decoding and core storage drive system.
- The binary address in DR-S, T, and R is displayed on the CPU console.
- DR-A is displayed on the CPU console in the 1 bit position of DR-E.

Description

To address a byte in Main storage, DR-A, DR-S and T are directly wired to the corresponding STAR position (Figure 2-14). The contents of DR-R is gated to the bus and from there to the four lower STAR positions.

Addressing the 4K Storage (Figures 2-14, 2-15, 2-15a and 2-16)

- Binary STAR addresses are split into two groups.
- The X drive lines are activated from the STAR positions 128, 256, 512, 1024, and 2048.
- The Y drive lines are activated from the STAR positions 1, 2, 4, 8, 16, 32, and 64.

Description

The numbers in parenthesis (-1-) in text refer to the encircled numbers in Figure 2-15 .

There are 128Y drive lines separated into a group of 16 Write Driver/Read Switches, with 8 lines to each of the 16 Drive/Switches (-1-). These 16 Write Driver/Read Switches are addressed by the DR-R 1, 2, 4 and 8 bits via STAR positions 1, 2, 4 and 8. See Figures 2-14 and 2-16 . The other end of the 128Y drive lines are connected to 8 Read Driver/Write Switches with 16 lines to each of the 8 (-2-). These 8 Read Driver/Write Switches are addressed by the DR-T 1, 2, and 4 bits via STAR positions 16, 32, and 64.

Note that in each of the 128Y lines current can flow in one or the other direction depending upon whether a Read (Fetch) or a Write (Store) operation is being performed.

There are 32 X drive lines separated into a group of 8 Write Driver/Read Switches, with 4 lines to each of the 8 (-3-). These 8 Write Driver/Read Switches are addressed by the DR-T, 8 bit and DR-S, 1 and 2 bits via STAR positions 128, 256, and 512. The 32 X lines thread through one half of a plane (4K) and are then jumpered (-4-) to return through the other half of the plane where they are connected in groups of 8 to 4 Read Driver/Write Switches (-5-). These four Read Driver/Write Switches are addressed by DR-S, and 8 bits via STAR positions 1026 and 2048.

Note that in each of the 32 X lines current can flow in one or the other direction depending on whether a Read (Fetch) or a Write (Store) operation is being performed.

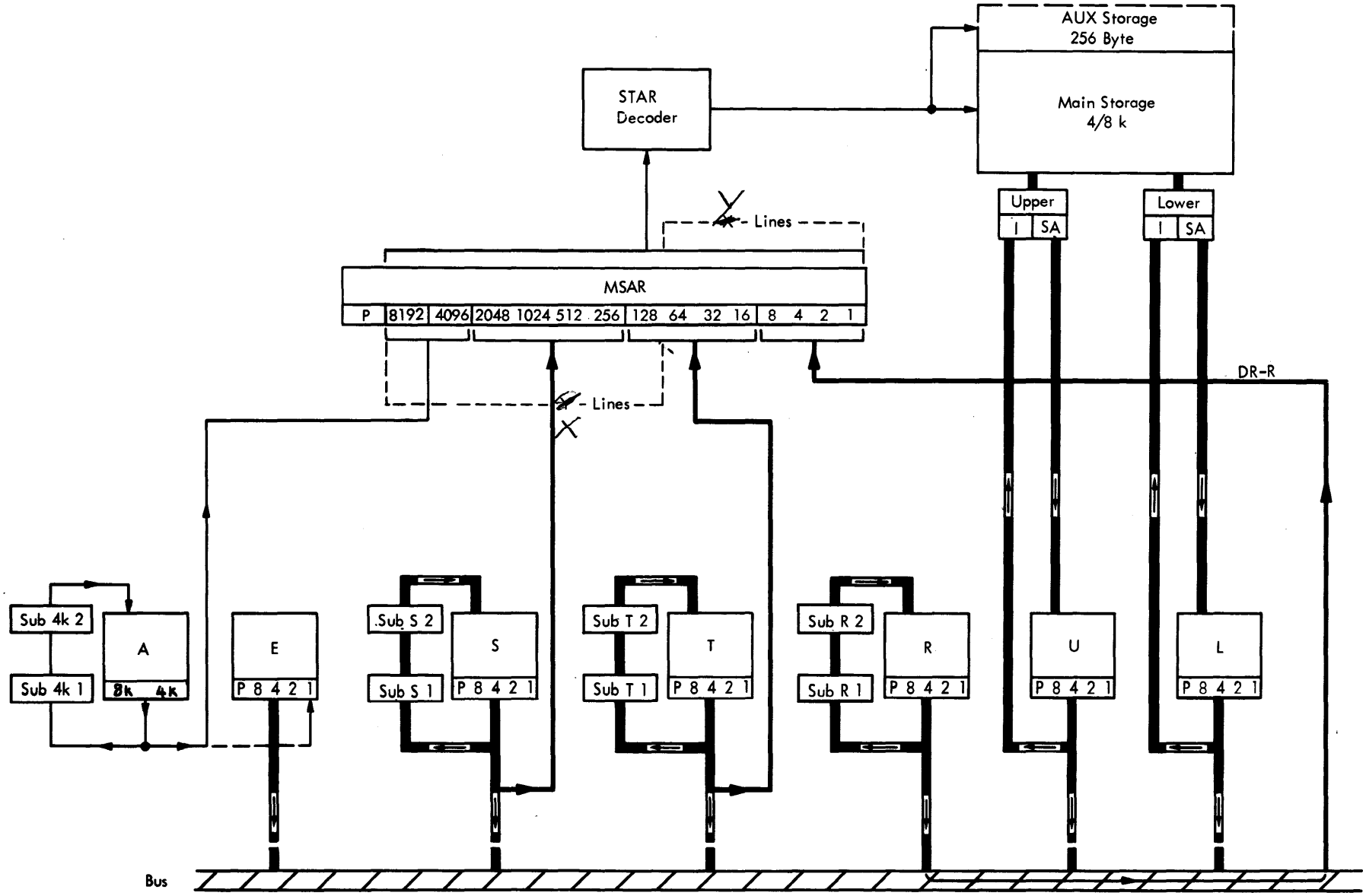


Figure 2-14 Main Storage Addressing System

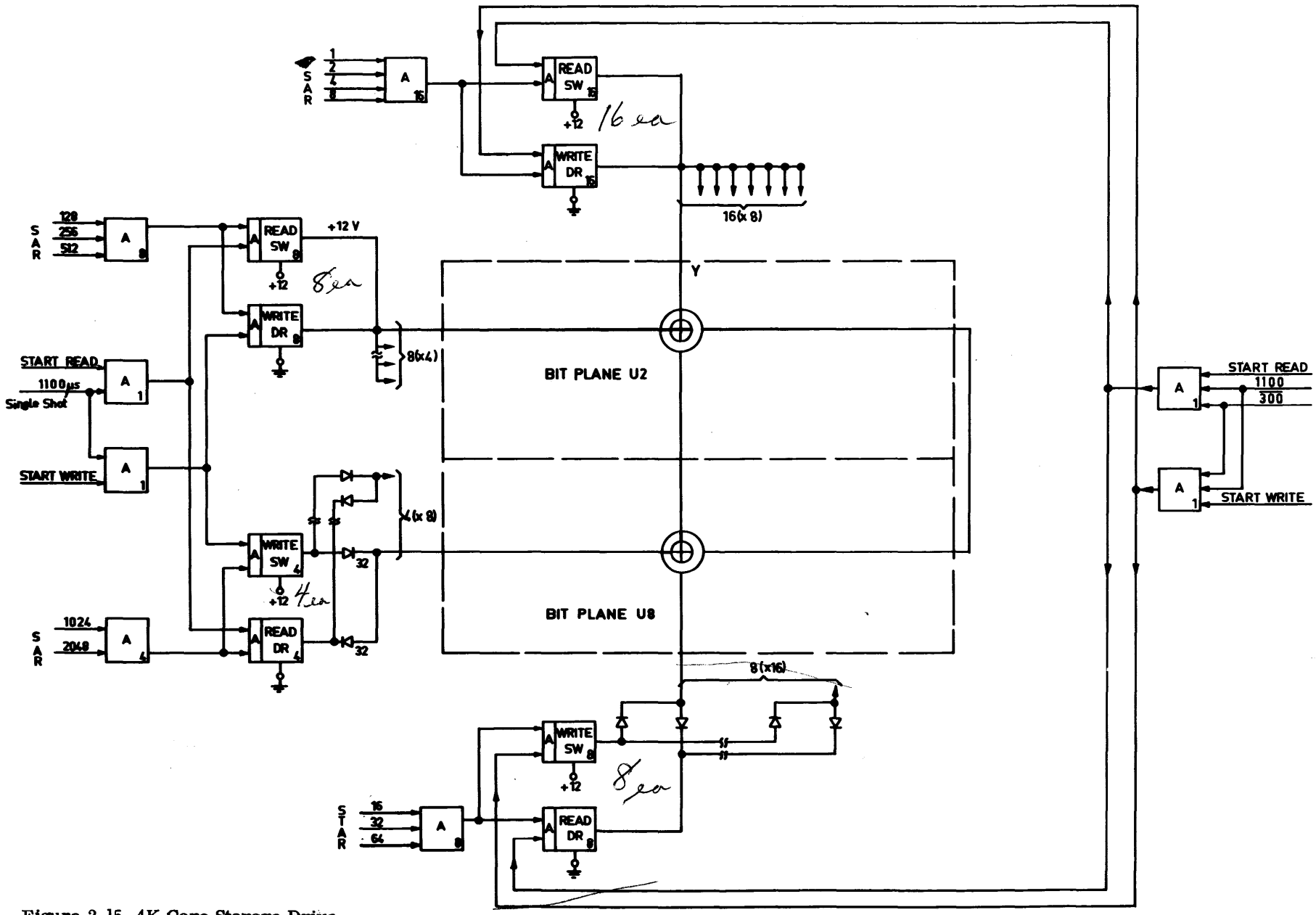


Figure 2-15 4K Core Storage Drive

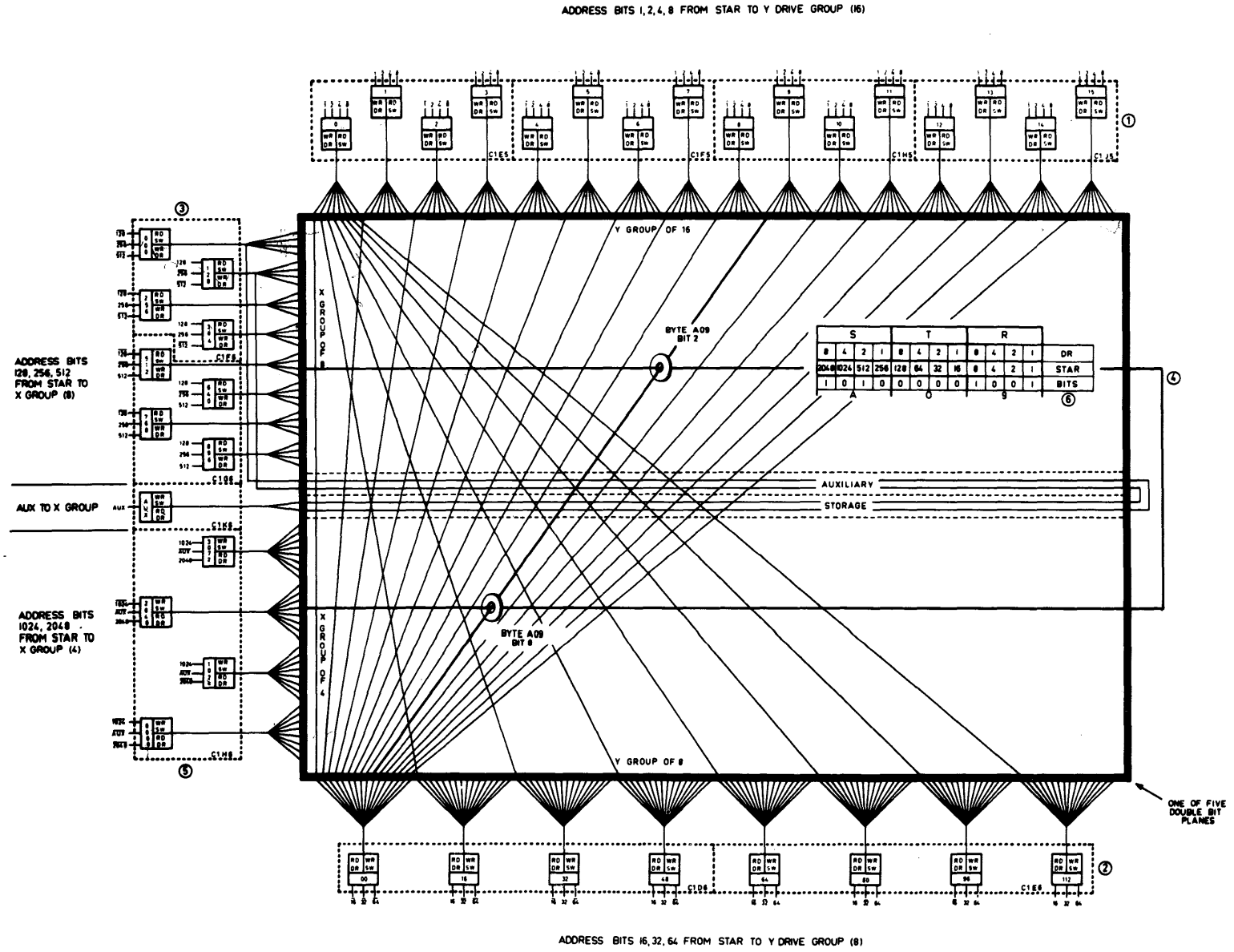


Figure 15a 4K Storage with 8K Planes

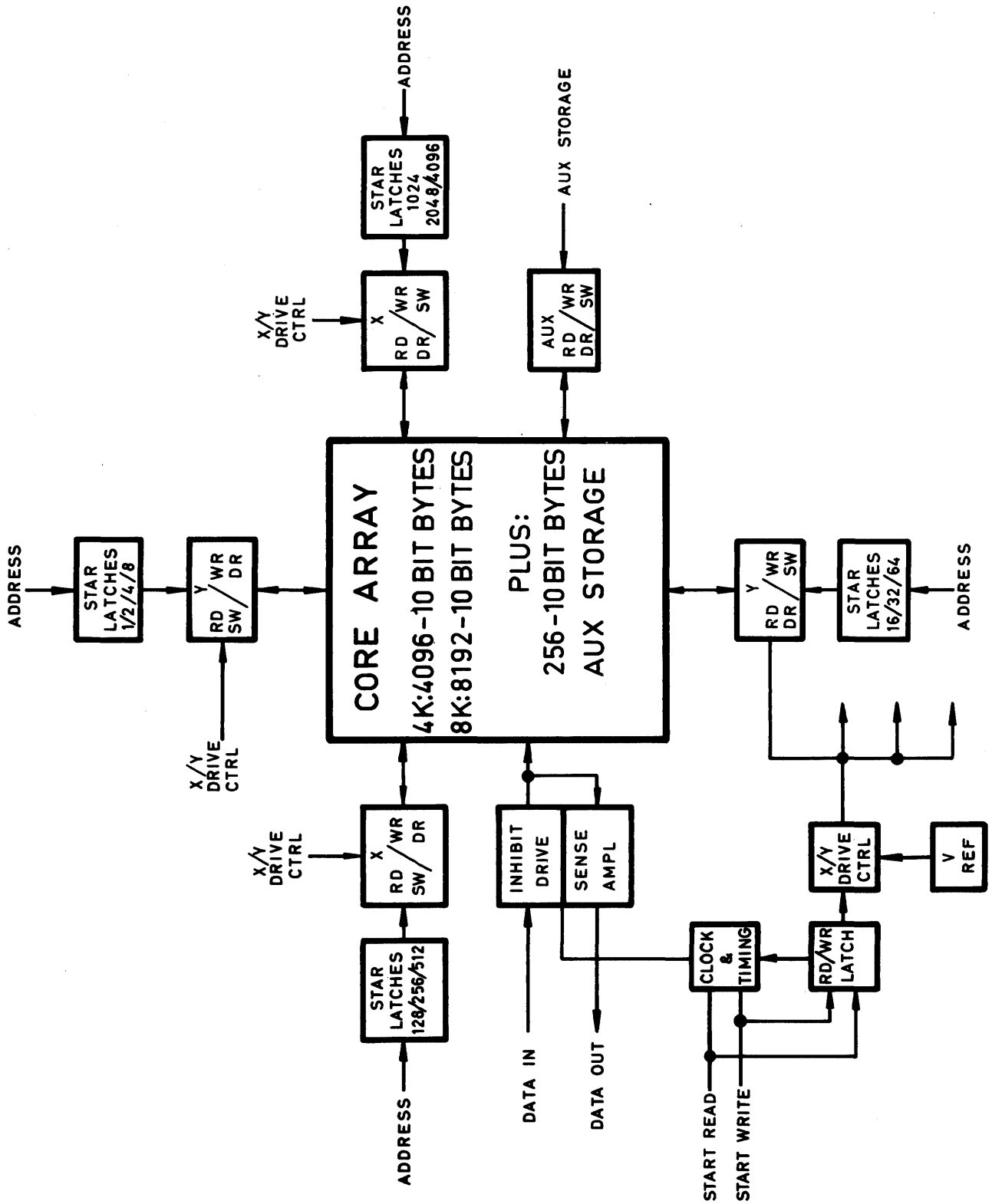


Figure 2-16. Principle 4/8K Core Storage Drive

Hexadecimal Notation

The hexadecimal notation for the binary address of the byte in storage is obtained by converting each of the four Data Registers into its hexadecimal notation as shown below.

DR-E	DR-S	DR-T	DR-R	Reg
P8421	P8421	P8421	P8421	Bit Pos
10000	11010	10000	11001	Binary
0	A	0	9	Hexadecimal

The example just given is also shown at (-6-) in Figure 2-15.

Main Storage

Driver/Switch Identification Driver/Switch blocks are numbered (identified) according to the binary sum in the STAR positions that address a block. For example, STAR positions 1, 2, 4, and 8 address the Y group of sixteen blocks (-1-) in Figure 2-15. The possible binary sums of 1, 2, 4, and 8 range from 0 (no bits) through 16 (all bits) in steps of 1, i.e., 1, 2, 3, etc. through 16. These STAR positions are displayed on the CPU console as DR-R 1, 2, 4, and 8 bits.

The STAR positions 16, 32, and 64 address the Y group of eight blocks. The possible binary sums of 16, 32, and 64 range from 00 (no bits) through 112 (all bits) in steps of 16, (that is, 00, 16, 32, 48, etc., through 112). These STAR positions are displayed on the CPU console as DR-T, 1, 2, and 4 bits.

The STAR positions 128, 256, and 512 address the X group of eight blocks. The possible binary sums of 128, 256, and 512 range from 000 (no bits) through 896 (all bits) in steps of 128, (that is, 000, 128, 256, 384, etc., through 896). These STAR positions are displayed on the CPU console as DR-T 8 bit and DR-S 1 and 2 bits.

The STAR positions 1024 and 2048 address the X group of four blocks. The possible binary sums of 1024 and 2048 range from 0000 (no bits) through 3072 (all bits) in steps of 1024 (that is, 0000, 1024, 2048, and 3072). These STAR positions are displayed on the CPU console as DR-S 4 and 8 bits.

NOTE: The X group of four blocks (0000 through 3072) also have a Not Auxiliary Latch function ANDed with STAR for addressing purposes.

Addressing the 8K Storage

- The only difference between the 4K storage and the 8K storage is in the doubling of the 32 X lines. See Figure 2-17.
- The addressing of the 64 X lines depends upon STAR positions 128, 256, 512, 1024, 2048, and 4096.
- STAR position 4096 is additional and is set depending upon the content of the DR-A.
- STAR positions 128, 256, and 512 are labeled as X group 8A (3 STAR positions give 8 address possibilities).
- STAR positions 1024, 2048, and 4096 are labeled as X group 8B (3 STAR positions give 8 address possibilities).
- X group 8A and 8B give the 64 addresses to select all 64 X lines.

Description

Figure 2-17 illustrates the distribution of the Driver/Switches in the 8K Storage.

Y Group of 16	Write Driver/Read Switches
X Group of 8(A)	
Y Group of 8	Write Switches/Read Driver
X Group of 8(B)	

Data Register A

The basic size of the Model 20 core storage is 4096 bytes. Three registers S, T, R for example, are sufficient to address 4096 bytes of storage. When all the bits in DR S, T, R are turned on, the highest possible address (4096) is addressed.

However, a larger core storage may be installed. An 8K storage or a 16K storage is available. To address an 8K storage, three data registers are not enough because one more bit is needed. Two more bits are needed to address a 16K storage.

These two additional bit positions are contained in Data Register A. DR-A consists of only two latches. One latch has the bit value of 4096 the other one has the bit value of 8192. DR-A also has two Sub-Registers holding 2-bits each for address exchange purposes.

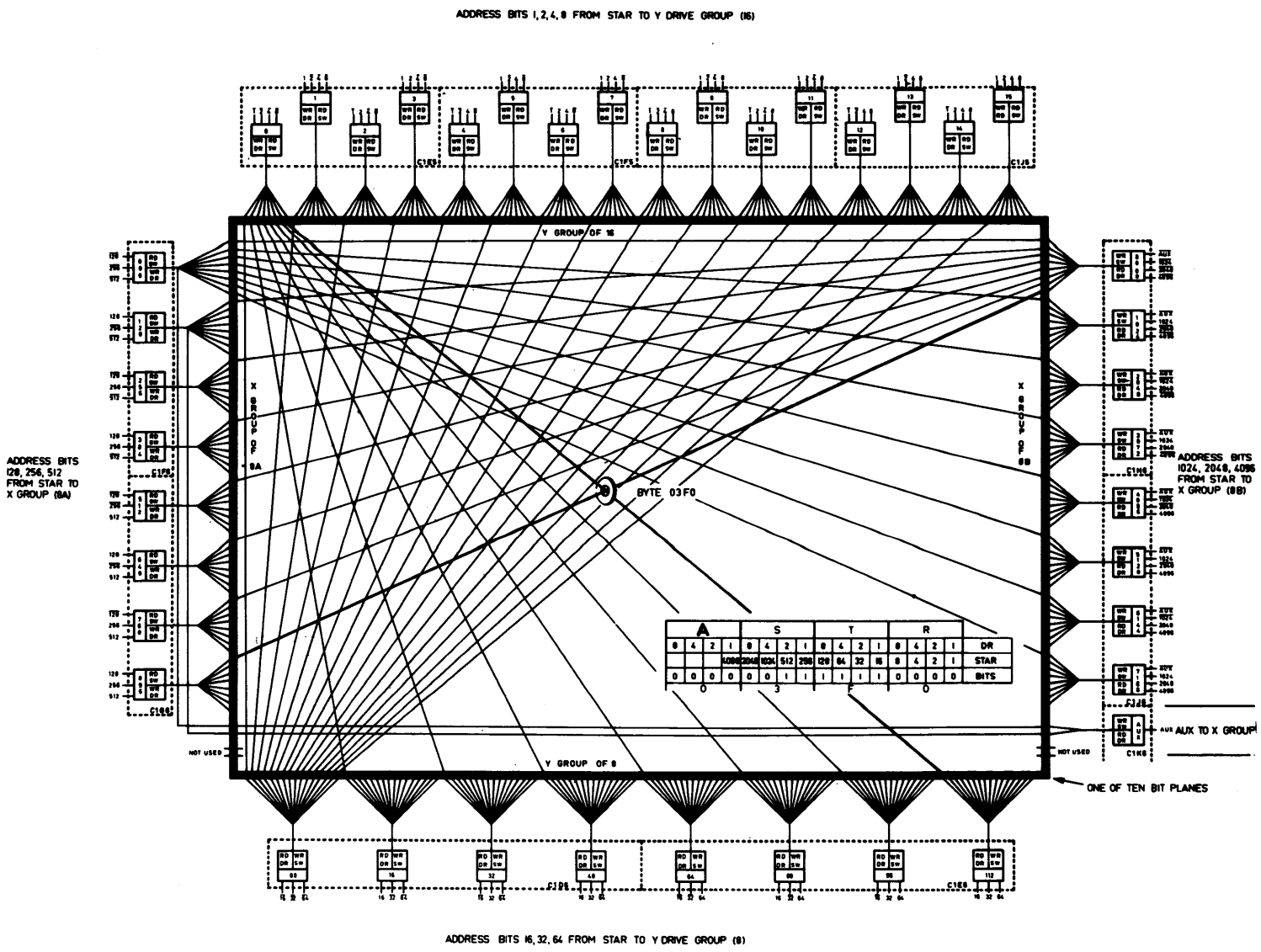


Figure 2-17 8K Storage with 8K Planes

Operation (Figure 2-18 and 2-18a)

When an address is read out of the Auxiliary Storage, the first portion is in DR U and L. To address STAR it is necessary to set up this address in DR S, T, R. Therefore, the content of DR-U is moved to DR- T and the content of DR-L is moved to DR-R. Thus 8 of the 13 bits are in DR-T and R.

The next portion of the Address is read into DR-U and L. For an Address not bigger than 4K, only DR-L contains data. The content of DR-L is then moved to DR- S and with that, the complete Address is in DR-S, T, R. With this last micro-instruction, that is MOVE L→S, one of the two bits in DR-A is set if DR-U contained either a 1 or a 2 bit. When DR-U contains a 1-bit, the 4K Address latch is set. When DR-U contains a 2-bit, the 8K Address latch is set.

The reverse operation is performed similarly. When an address which is set up in DR-A, S, T, R is stored back into Auxiliary Storage (for example) the content of DR-A is automatically set into DR-U by a Move S→L micro-instruction (provided that the latches in DR-A are turned on).

STORAGE ADDRESS REGISTER (STAR)

- STAR is a latch register capable of containing one complete Core-storage address.
- STAR receives the Core-storage address from various DR's depending upon which micro-instruction is used to address (activate) Core-storage.
- For a 4K storage, STAR has 12 positions. For an 8K storage STAR has 13 positions. For 16K storage STAR has 14 positions.
- For a F/S N micro-instruction, the 4 bit N part of ROSDR position 6-9 is set into STAR.
- For a F/S PN micro-instruction, the content of DR-P and the 4 bit N part of ROSDR position 6-9 are set into STAR.
- For a F/S STR micro-instruction, the content of DR A, S, T, and R is set into STAR.
- Figures 2-14 and 2-19 show the Data Paths for addressing Core-storage.
- The content of STAR is decoded in the STAR Decoder to address the proper byte in Core-storage.

Description

To read a byte out of Core-storage (Fetch) or to write a byte into Core-storage (Store), the Core-storage must be addressed. The address of a byte in Core storage is, with Clock A, set into the STAR at the beginning of a Fetch or Store operation (micro-instruction).

When the address which is set into the STAR comes from the N part of ROSDR (4 bits) or from the N part combined with DR-P (8 bits), the remaining STAR positions are reset-set to their off (zero bit) status.

AUXILIARY STORAGE

- Auxiliary (AUX) storage is physically but not electrically, a part of the core storage unit.
- Except for the AUX latch, the same addressing, read in and read out circuits (STAR Decoder) are used as for main storage (Figure 2-20).
- AUX contains 256 addressable bytes (0-255). (Figure 2-21).
- To address AUX storage an eight bit address with a value of 0 - 255 is set into STAR.
- As shown in Figure 2-22, STAR receives the 8 bit address from:
  1. The "N" part of the ROSDR (Positions 6-9, 4 bits).
  2. The 4 bits in Data Register P.
- AUX storage addresses (0-255) are separated from Main storage address (0-255) by the addition of a latch, the AUX latch (Figure 2-19).
- AUX latch 1 is set by FN, SN, FPN, and SPN micro-instructions (Figure 2-23).
- FN and SN micro-instructions can select only row 0 (16 BYTES) (Figure 2-24).
- FPN and SPN micro-instructions can select any of 256 BYTES.
- All data from or to AUX storage passes through Data Registers U and L (Figure 2-20.).
- AUX latch 2 is an optional feature used with the IBM 2520.

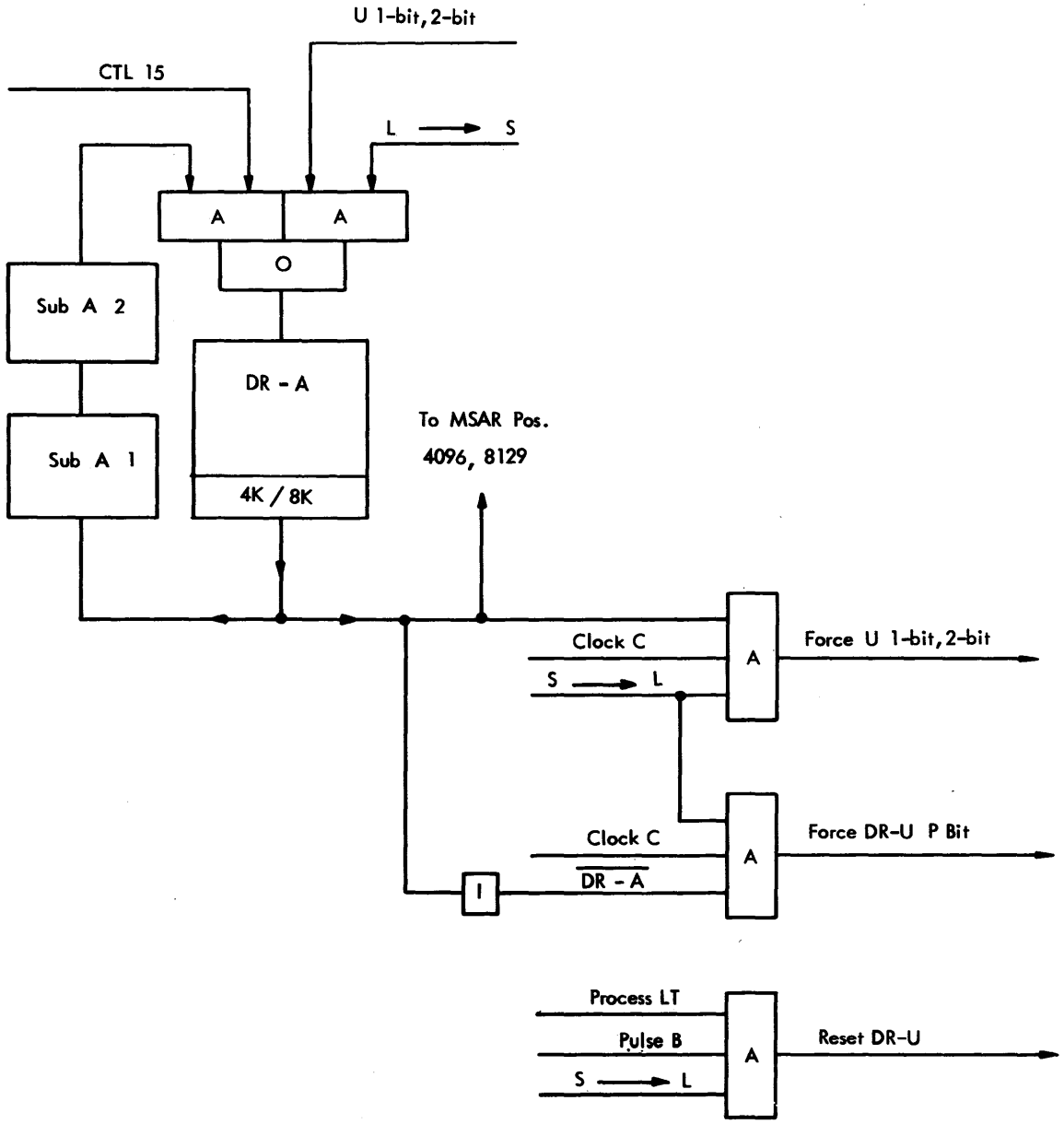


Figure 2-18 Data Register A

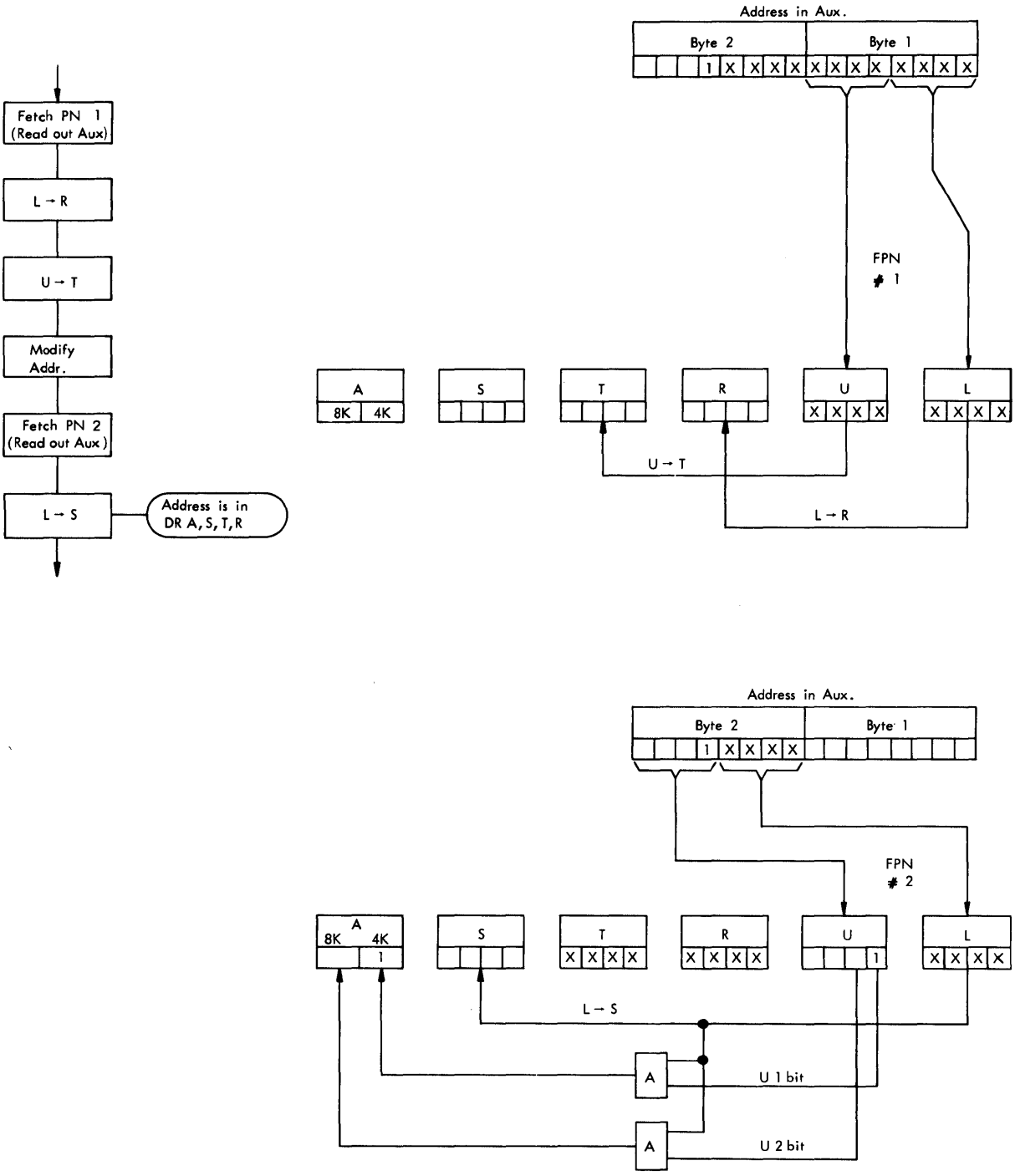


Figure 2-18a Address Transfer with Setting of DR-A

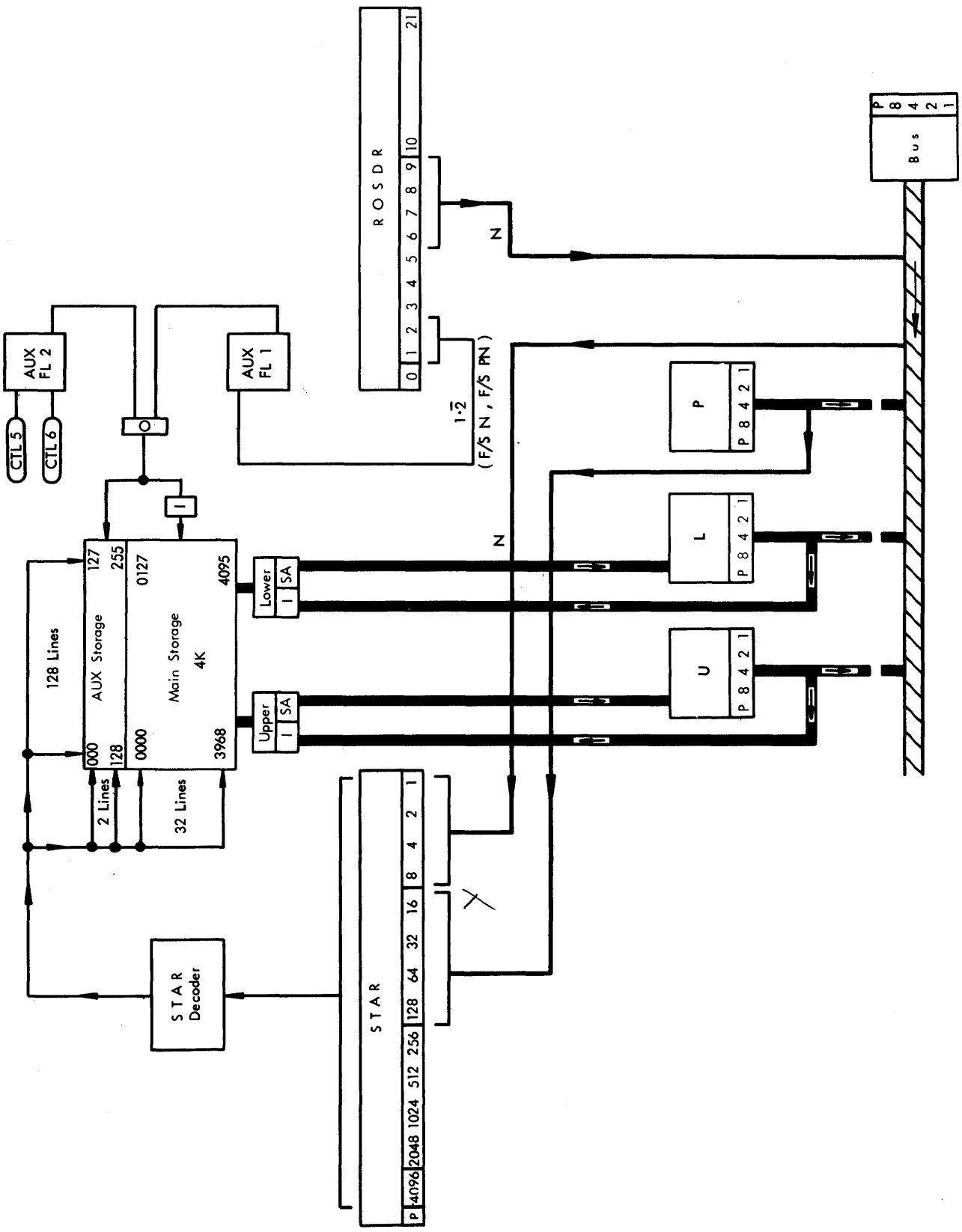


Figure 2-19. Data Path to Address Aux Storage

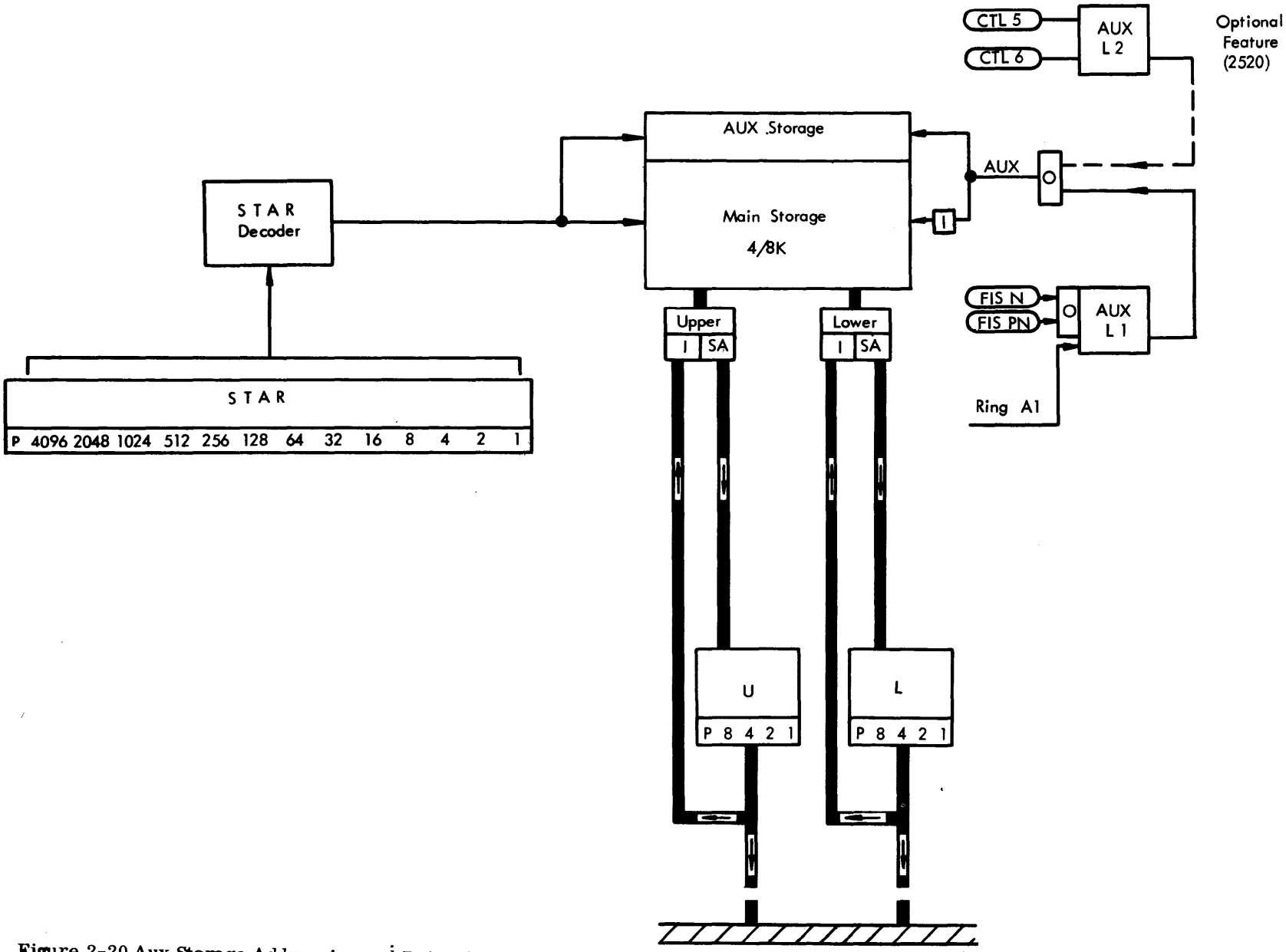


Figure 2-20 Aux Storage Addressing and Data Flow

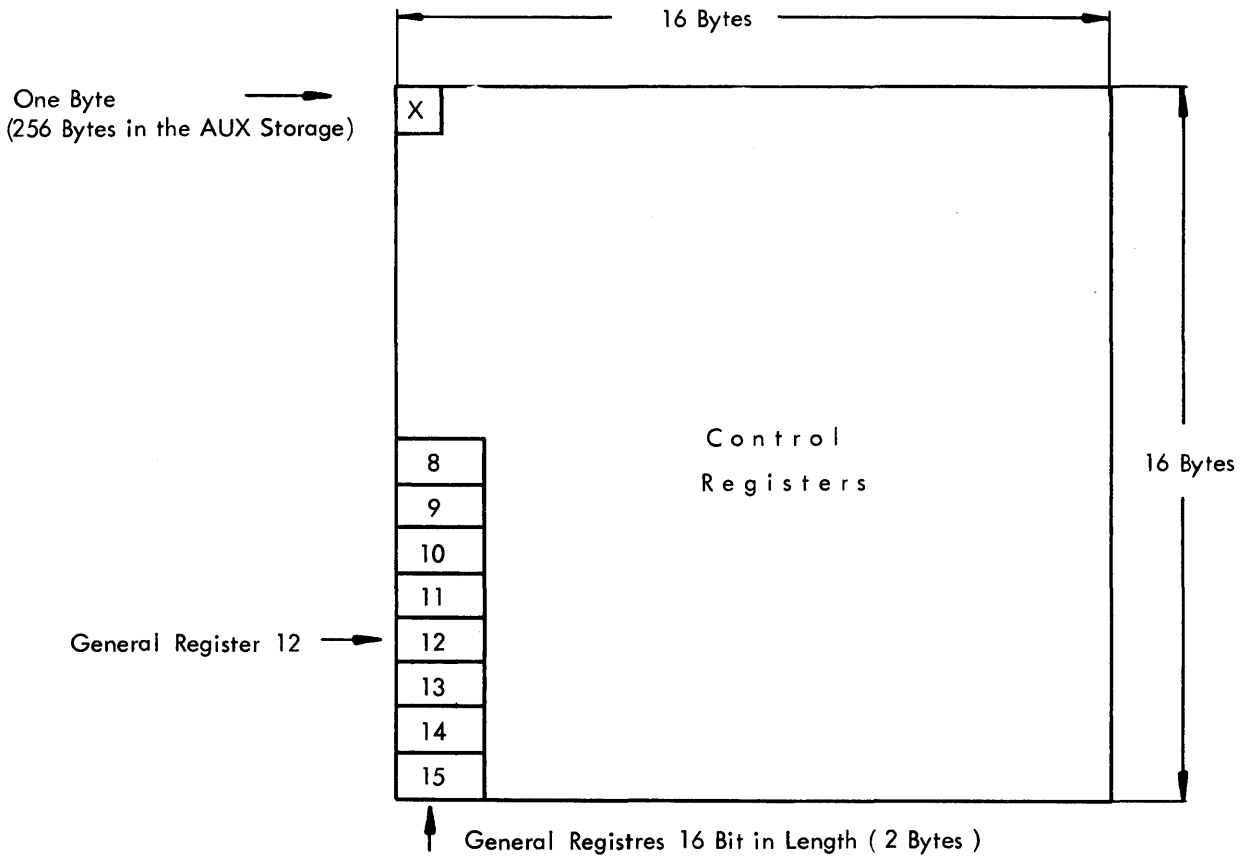


Figure 2-21 Aux Storage Content

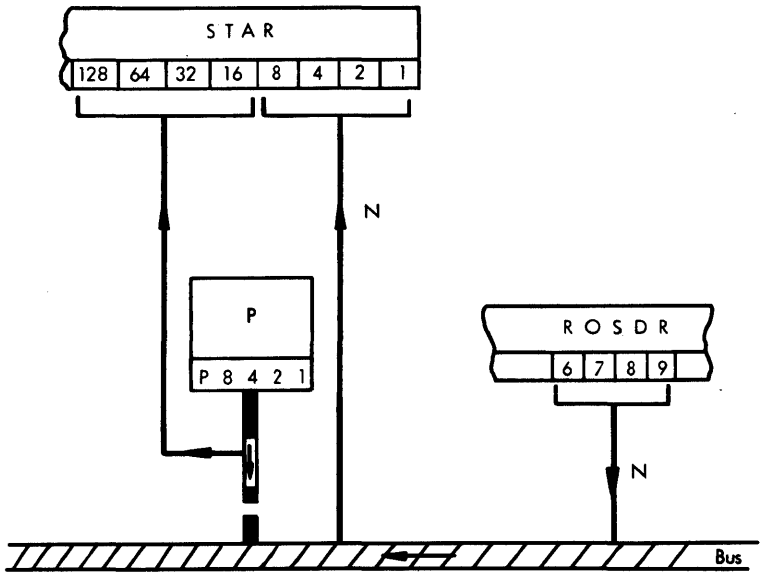


Figure 2-22 Aux Storage Address to STAR

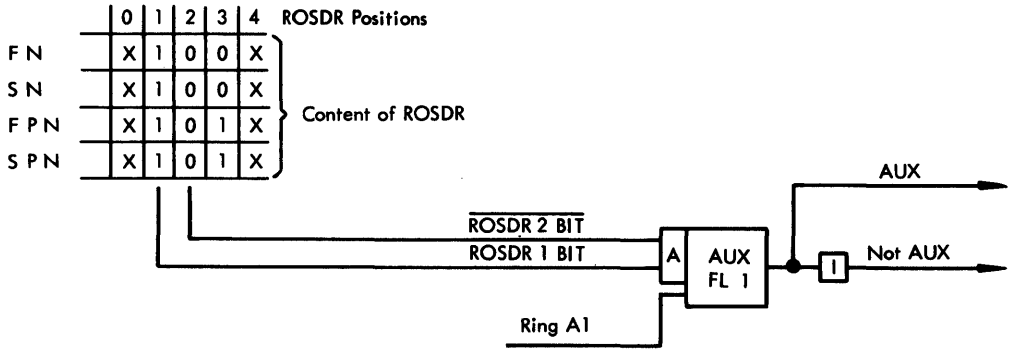


Figure 2-23 Setting Aux Latch 1

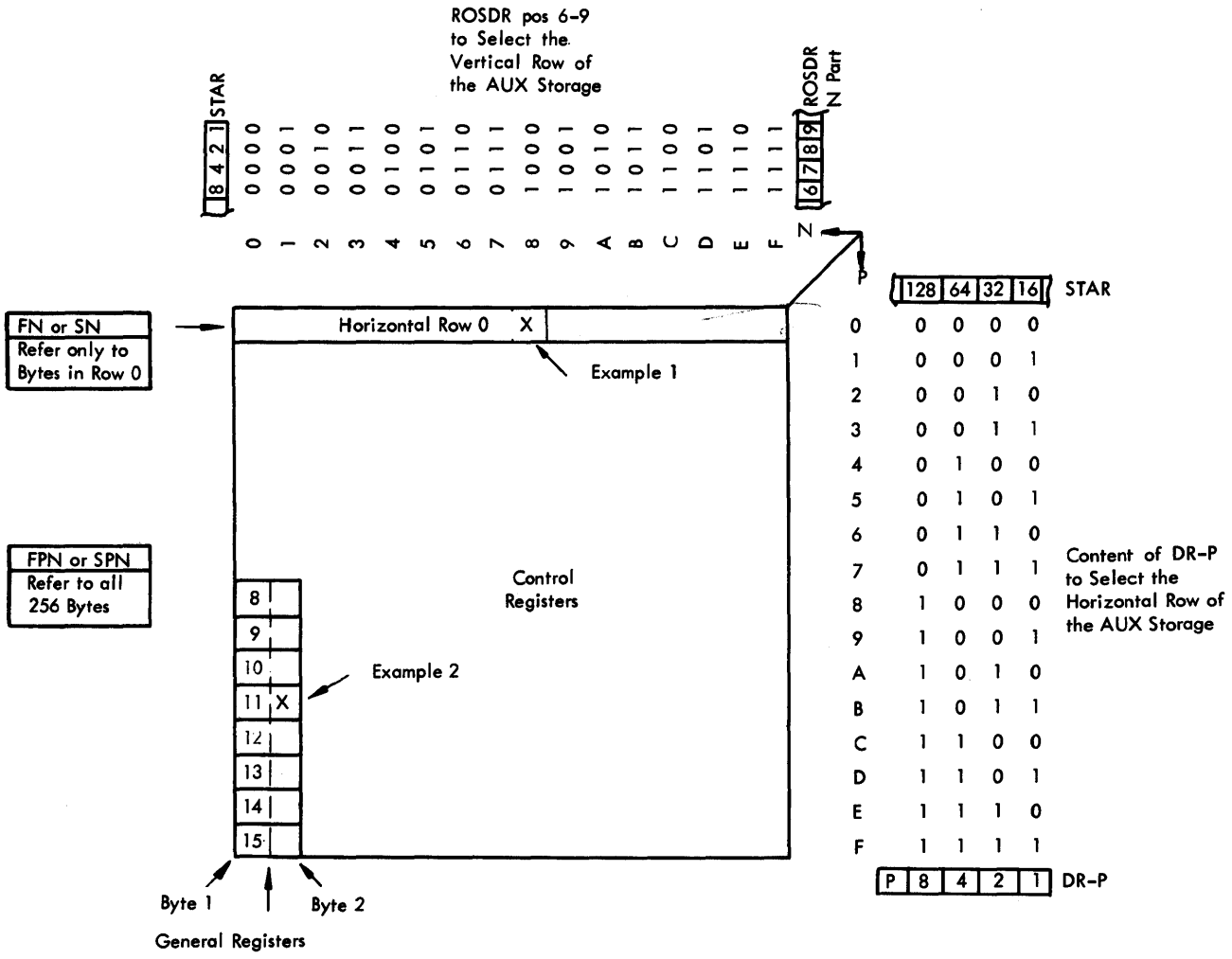


Figure 2-24 Addressing of Aux Storage Bytes by Micro-instructions

Description

Figure 2-20 shows the true location of the AUX storage in reference to main storage.

Addresses from 0-255 in STAR can select a byte either in Main storage or AUX storage. The AUX latch is used to distinguish between AUX storage and Main storage. The AUX latch is set (ON) when AUX storage is used (referred to) by any one of four micro instructions: FN, SN, FPN, and SPN. All four of these micro instructions have a 1 bit in ROSDR position 1 and a 0 in ROSDR position 2 ($1 \cdot \overline{2}$). The $1 \cdot \overline{2}$ condition sets the AUX latch (Figure 2-23).

When the AUX latch is not set (OFF) and an address of 0-255 is in STAR, positions 0-255 in Main Storage are addressed.

All data from or to AUX storage passes through Data Registers (DR-U and L. The same applies to Main Storage data. The N part of the FN, SN, and SPN micro instructions address the vertical rows (0-F) in AUX storage as shown in Figure 2-24. The P part of the FPN and SPN micro instruction addresses the horizontal rows (0-F) in AUX storage as shown in Figure 2-24. An FN or SN micro instruction having only a 4 bit N address, can select only one of 16 bytes in the horizontal row 0 of AUX storage.

The hexadecimal addresses of these locations are /00-0F/. An FPN or SPN micro-instruction having a 4 bit N address and a 4 bit P address can select any one of the AUX storage bytes up to 256.

During an FN or SN micro instruction operation, the "N" address (position 6, 7, 8, and 9 of ROSDR) is set into STAR positions 1, 2, 4, and 8. All other STAR positions are set to binary zeros (Figure 2-19).

During an FPN or SPN micro instruction operation the "N" address is set into the main STAR positions 1, 2, 4, and 8, and the Data Register P content is set into STAR positions 16, 32, 64 and 128. All other positions of STAR are set to binary zeros (Figure 2-19).

Two examples of addressing AUX storage are given here (Figure 2-24).

Example 1: An FN or SN micro instruction selects BYTE 8 in horizontal row 0 when the bit pattern of the "N" part in ROSDR is 1000. This address selects vertical row 8 (Hexadecimal position /08/).

Example 2: An FPN or SPN micro instruction refers to the second byte of General Register 11 when Data Register "P" contains 1011 which selects horizontal row 11 and the bit pattern of the "N" part in ROSDR is 0001 which selects vertical row 1 (Hexadecimal position /B1/).

Auxiliary Storage Content

- Auxiliary storage contains general registers and Control Registers (Figure 2-21).
- The 8 general registers are machine program addressable and are numbered 8-15.
- The general registers contain either Fixed Point Data or Base Addresses.
- The quantity of the Control registers in use depends on the number of attached I/O units.
- Control registers are used as OP register, address register or I/O Buffers.

TRANSFORMER READ ONLY STORAGE (TROS)

- TROS is a storage unit which contains fixed predetermined information (Micro-instructions).
- The information can be read out only.
- Alterations to the stored information can be made only by physical changes to the TROS hardware (Tapes).
- The capacity of TROS is 2K words which represents 6144 micro-instructions.
- A 13-bit Read Only Address Register (ROAR) allows the addressing of the 6144 micro-instructions.
- A micro-instruction which is read out of the TROS is set into a Read Only Storage Data Register (ROSDR).

Description

When a current pulse is passed through the primary winding of a transformer, it will induce a current pulse in the secondary winding. No primary current pulse (or winding) results in no secondary output. This is the principle of TROS (Figure 2-25).

The primary of the transformer is a drive line which is addressed from ROAR. The secondary of the transformer is the sense winding. The sensed value (bit) is set into ROSDR.

When a drive line runs through a transformer core, a current pulse in the drive line induces a current pulse in the secondary winding.

If the same line by-passes a transformer core then no current pulse is induced in that particular sense winding. A pulse in the sense winding represents a one bit, no sense winding output represents a zero bit. Additional drive lines could be used in a similar manner. An example is given in Figure 2-26.

1. A current pulse in drive line A gives an output of 101.
2. A pulse in drive line B gives an output of 011.

With 3 transformers, an output of 3 bits results. With N transformers, an output of N bits results. With 2 drive lines, 2 different bit configurations are obtained. With M drive lines and N transformers a storage capacity of M words, each word consisting of N bits, is obtained.

Purpose and use of TROS. The content of any TROS word can be read out and placed in the Read Only Storage Data Register (ROSDR). The register information (Micro instructions) can be decoded and used to control machine functions. In addition, part of the ROSDR output, which may be modified by machine condition bits, determines the address of the next TROS micro-instruction.

TROS micro instructions are addressed in a particular sequence. The sequencing of TROS addresses is called a Microprogram. To perform any operation in the machine, the various components of the CPU (Main Storage, Registers and Modifier, etc.) are controlled by microprogram to perform certain functions in a certain sequence.

Thus, an operation is defined by a sequence of TROS words, which is the microprogram for a particular operation.

TROS Physical Description

- TROS is built up of modules containing 256 TROS words.
- Each word consists of three micro-instructions which have a length of 16, 22, and 22 bits.
- When a micro-instruction is addressed, a whole word is always read out but only the addressed micro-instruction is used (Set into ROSDR).
- There are 768 micro-instructions in one module so eight modules are necessary to hold all 6144 micro-instruction.

Description

Transformer. The core of a TROS transformer consists of two parts, a U piece and I piece, (Figure 2-27). Both the U piece and the I piece are made of soft ferrite having low remanence. The U and I pieces are first coated with an insulating material and then copper plated. This is to reduce flux leakage. To prevent the plating from acting as a short circuited turn on the transformer, U cores are gapped around their outside face. A sense winding of 35 turns is wound around the I piece.

Tapes. To overcome the problems of winding the 256 drive lines contained in each module and to improve the electrical characteristics, the TROS drive lines are etched in copper on flexible plastic tapes. On each tape, two drive lines are printed, both in the form of a ladder network. Holes are punched between the rungs of the ladder so that U cores can be inserted through the tapes to mate with the I cores. Each leg of the U core is encircled by the sides of the ladder network and two of its rungs, so by interrupting one or the other side of the ladder the conductor may by-pass or link with the U core. In position link a "one" is programmed, in position by-pass a "zero" is programmed, (Figure 2-28. Figure 2-29 shows that the two drive lines terminate at the tab end of the tape. Each line represents a word (3 Micro-instructions). The outer conductor loop is called the A word, the inner loop is the B word. The bit positions along the tape are numbered 0-59. Bits 0 and 1 are at the tab end.

Physical Construction of the Module. An exploded view of a module is shown in Figure 2-30. The tape deck (18) consists of 128 plastic tapes which carry

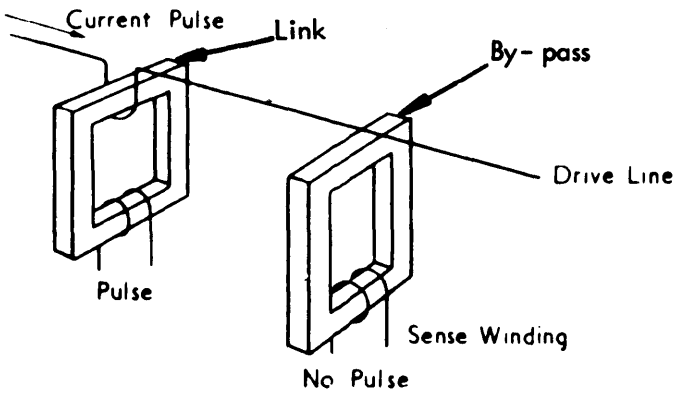


Figure 2-25 Principle of TROS

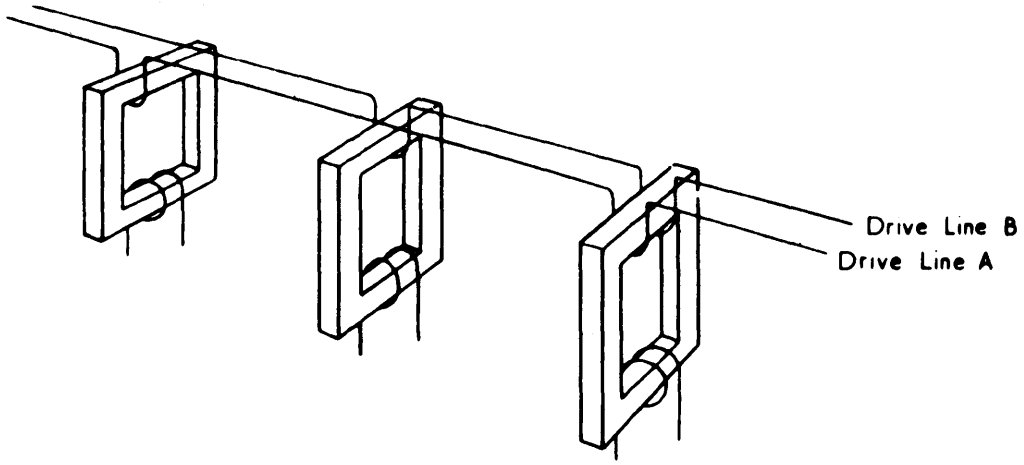


Figure 2-26 Simplified TROS

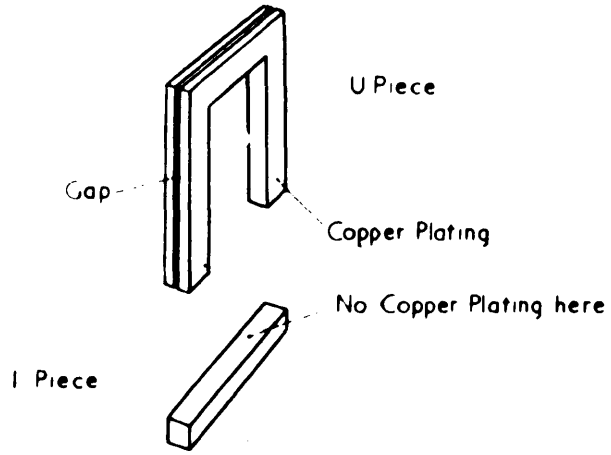


Figure 2-27 TROS Core

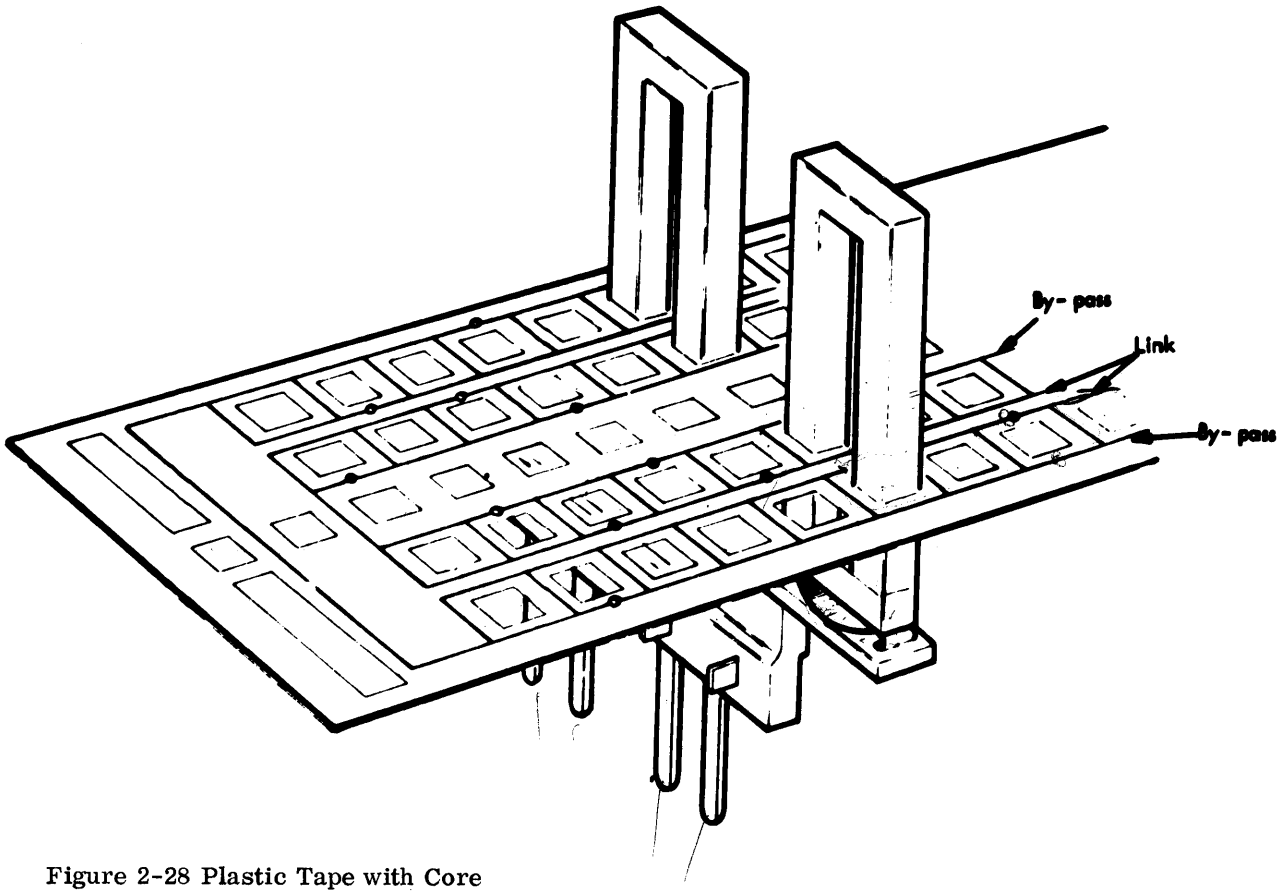


Figure 2-28 Plastic Tape with Core

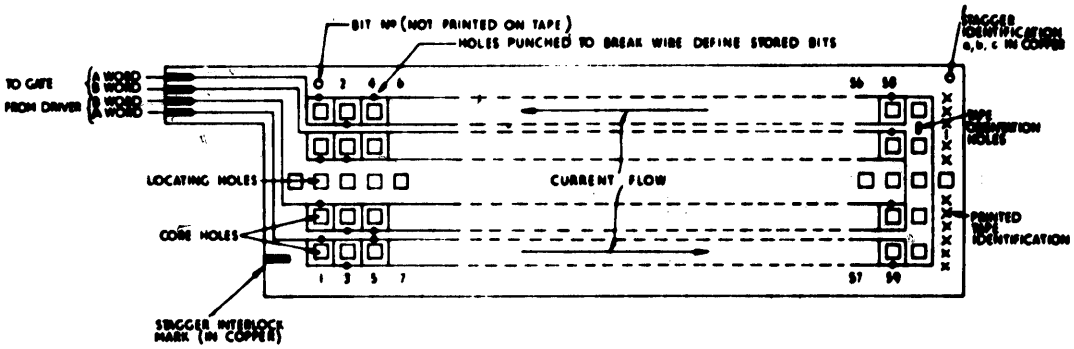


Figure 2-29 Tape Layout

1. Rod
2. Block
3. Carrier
4. 'I' Core
5. Spring
6. Strip
7. Chassis
8. Support
9. Rail
10. Diode Board
11. FDD Substrate
12. I - O Cables
13. I - O Cables
14. I - O Cable Card
15. Cable Clamp
16. Terminating Pins
17. Clamp
18. Tape Stack
19. Tape Stack
20. Alignment Pin
21. 2 Banks of 30 'U' Cores
22. Retainer
23. Insulator

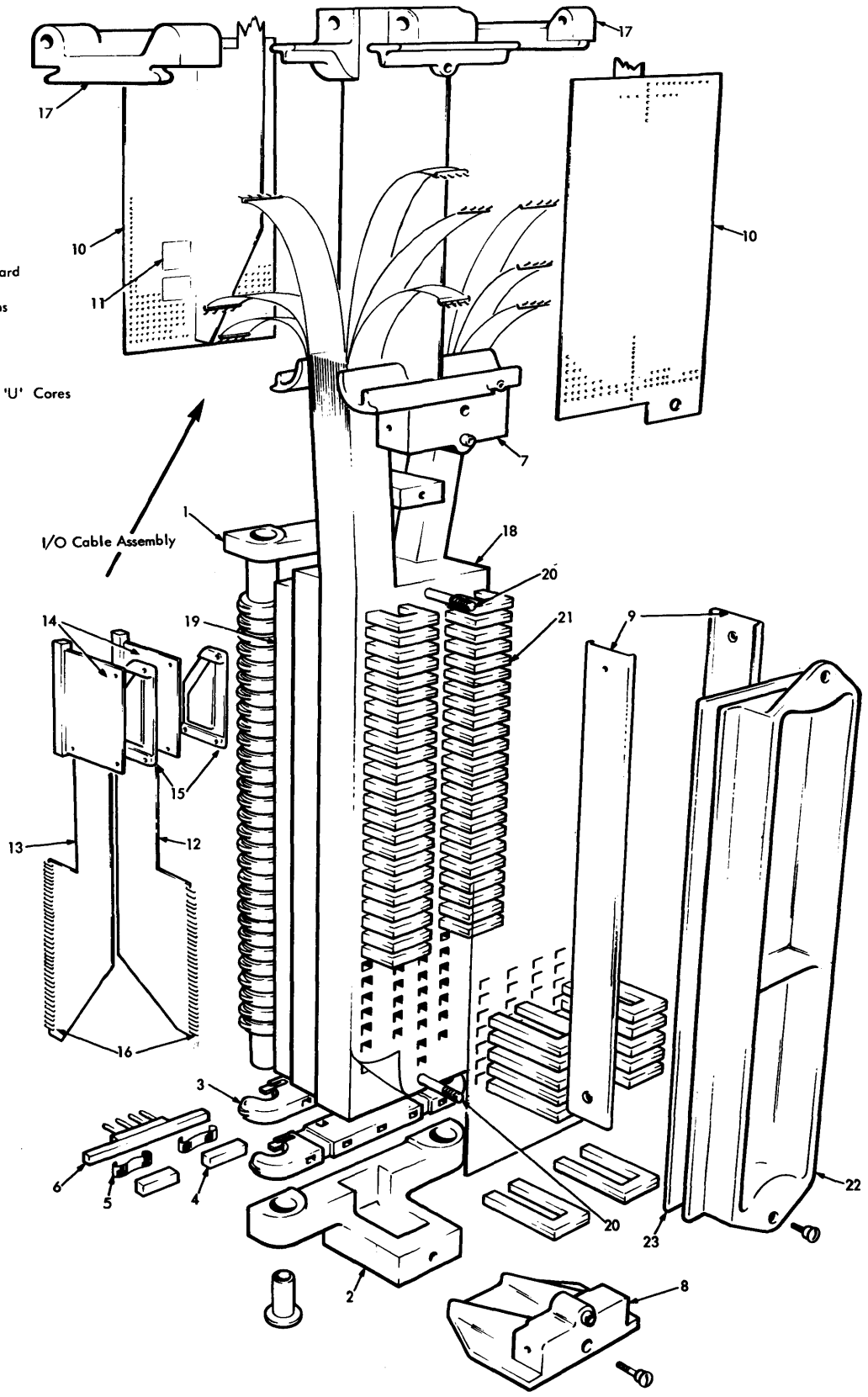


Figure 2-30 Exploded View of a TROS Module

the 256 words. Holes are punched between the rungs of the ladder network to accept the U cores (21) which pass through the tapes and mate with the I cores (4). The I cores (4) are held in a core carrier assembly consisting of parts (3), (5) and (6). The I cores are wound with a sense winding of 35 turns, which is connected to the pins on part (6). The tapes are lifted on and off the module by the tray (19) and located by means of the aligning pins (20) which screw into the blocks (2). The blocks also carry the two rods (1) on which the core carrier assemblies clip. There are thirty core carriers and one dummy core carrier. The dummy core carrier (24) is next to core carrier (3) and is only used to support the tape deck. The support (8) and chassis (7) screw into the blocks (2) and are spaced by the rails (9). Connected to the chassis are the module end boards (10) which carry the diodes (11) used for TROS word addressing and connections to the tapes in the tape deck. The connections to the tape consist of pins which are placed in plated through holes in the board and soldered to the printed circuitry.

Input/Output connections to the module end boards are made by the tapes (12) and (13). These have pins (16) similar to those on the tapes in the tape deck, which pass through plated holes in the boards (10) and are soldered to the printed circuitry. The tapes are clamped to the chassis (7) by the clamps (17) to relieve any strain in the connections to the boards (10). The flexible ends of the tapes are connected to cards (14) and clamped with reliefs (15). The U cores (21) are held in the module by the retainer (22) and insulator (23). The retainer screws into the support (8) and chassis (7), forcing the U cores against the I cores (4) and compressing the spring (5).

Numbering of Tapes in the Tape Deck. The upper 64 tapes in the module are in an inverted position with respect to the lower 64 tapes (Figure 2-31). This is done to create more space for the connection of the tape ends to the module and boards. The bottom of the module is defined on the side nearest to the I cores. The tapes in the lower half of the tape deck are numbered 0 to 63 from the bottom up. The tapes in the upper half of the tape deck are numbered 64 to 127 from the top down.

TROS Addressing Principles

- To address any word line, a drive circuit is needed at one end of the line, and a gate circuit at the other end.

- To select (for example) any one of eight word lines (Figure 2-32. one of the 2 drivers and one of the 4 gates are selected.

Description

In order to select any one of the 2048 drive lines of the TROS (2048 Words), 64 drivers, 32 gates and associated diodes are needed. This matrix of drivers and gates is controlled by the decoded output of the 13 bits in the Read Only Address Register (Figure 2-33). The control of the 32 gates is by means of bits E, 10, 11, 12, 13 of the ROAR (5). The control of the 64 drivers is by means of bits 14, 15, 18, 19, 20, 21 of ROAR . There are 4 gates to every Module. The drivers are common to all 8 Modules (Figure 2-33).

ROAR positions 16 and 17 select the addressed micro-instruction (one of three), out of the word which is read out by activating the gates and drivers.

Circuit Description

ROAR position E and 12 are pre-decoded in Gate Pre-Decoder 1. ROAR positions 10, 11, and 13 are pre-decoded in Gate Pre-Decoder 2. The output of Gate Pre-Decoders 1 and 2 is further decoded in the Final Gate Decoder in gates 0 through 31. Every gate is connected to the 64 drivers by way of the primary winding of the TROS tapes. Only one of the 32 gates is activated to address a word (3 micro-instructions).

ROAR positions 14, 15 and 18 are predecoded in Drive Pre-Decoder 1. ROAR positions 19, 20, and 21 are pre-decoded in Driver Pre-Decoder 2. The output of Drive Pre-Decoder 1 and 2 is further decoded in the Final Drive Decoder in driver 0 through 63. Every driver is connected with diodes to each of the 32 gates by way of the primary winding of the TROS tapes. Only one of the 64 drivers is activated to address a word (3 micro-instruction). The drivers are timed with the Driver Strobe to activate the selected primary winding (word) of the TROS.

ROAR positions 16 and 17 are decoded and timed with the sense strobe signal to gate a third of the word (selected micro-instructions) which is read out of ROS into the ROSDR. Three different sense strobe signals are generated to gate one of the three micro-instructions into the ROSDR.

The bits in positions 16 and 17 of ROAR, the associated Sense Strobe signals as well as the bits of the word which are set into ROSDR are shown below.

ROAR Position Bits 16	ROAR Position Bits 17	Sense Strobe	TROS Word Bits
Zero	Zero	A	0-15
Zero	One	A	0-15
One	Zero	B	16-37
One	One	C	38-59

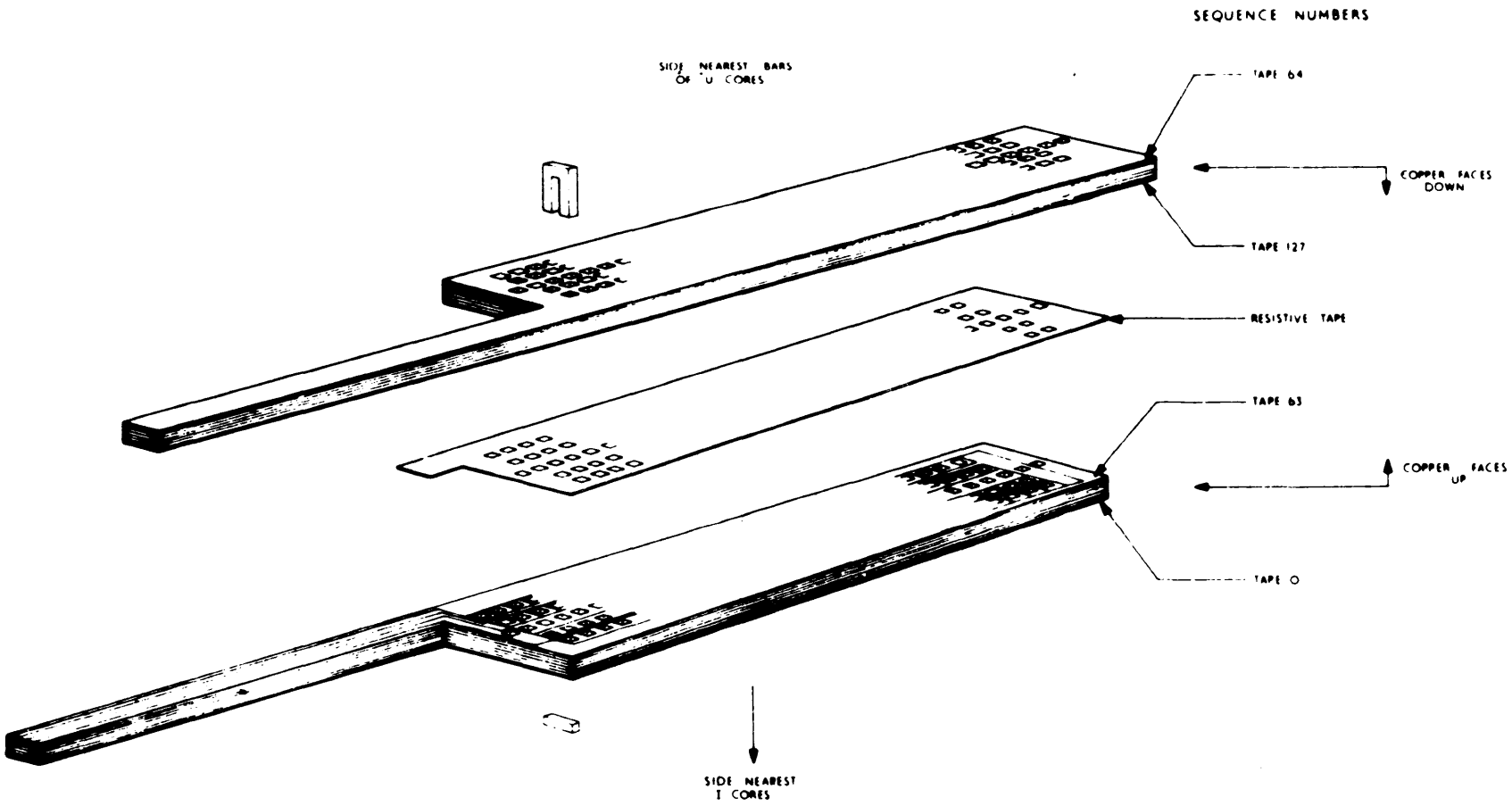
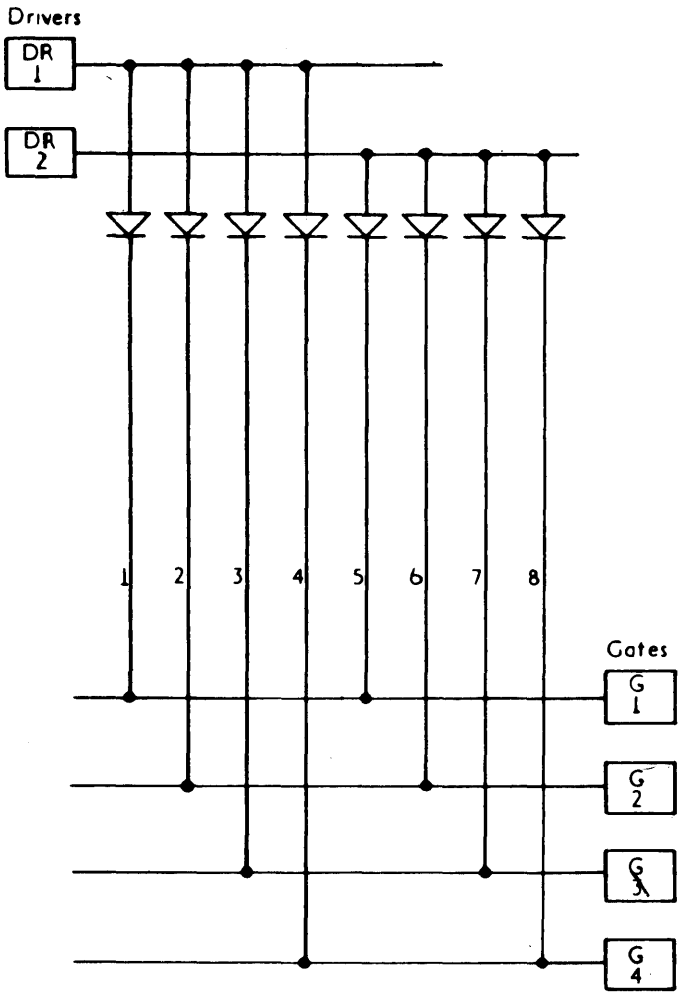


Figure 2-31 Numbering of Tapes in a Module



A Word Line is selected by using a driver and a gate. For example, by activating driver No2 and gate No 1, Word Line No 5 is selected and TROS Word No 5 is read out.

Figure 2-32 Principle of Driving and Gating

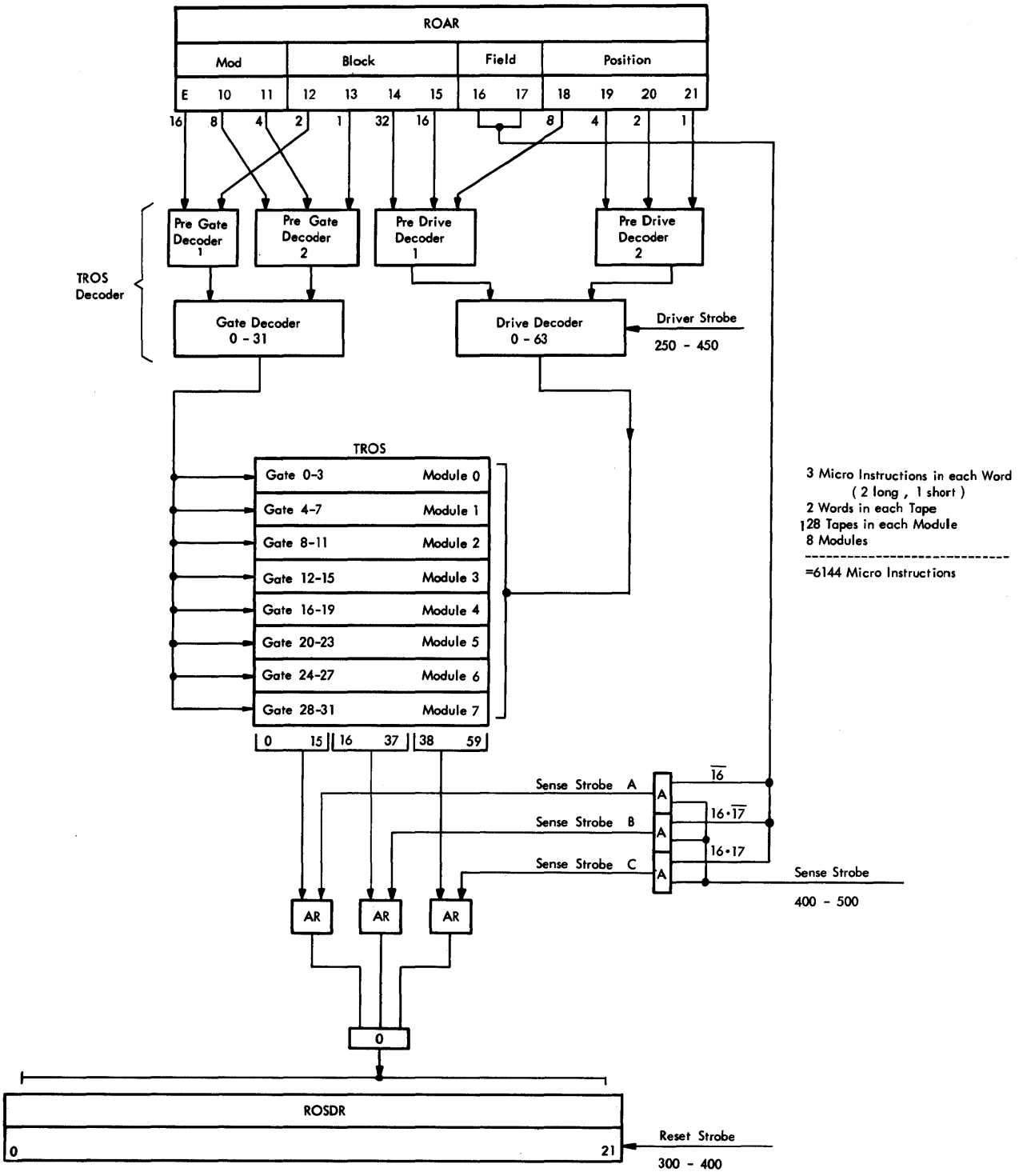


Figure 2-33 TROS Addressing Principle

ROAR OPERATION

- Clock A (from 0-250) resets the ROAR and sets the NSI address from the ROSDR into the ROAR if the ROAR-Advance signal is up.
- The ROAR Advance signal is controlled by the Ring latches.
- The Advance signal is normally present in every cycle except when a Fetch or Store micro-instruction is read out of the Read Only Storage.

Description (Figures 2-34, 2-35, 2-36 and 2-37)

A Fetch or Store comprises a problem because ROS cycle and core storage cycle are not in synchronism.

Fetch and Store Micro-instruction. On a Fetch and Store operation two different types of cycles occur:

1. The core storage cycle which takes 1.8 usec.
2. The ROS cycle which takes 0.6 usec.

Thus, a Read or Write cycle is three times as long as a ROS cycle. Therefore, the ROAR is not advanced during a Fetch or Store operation and no new NSI address is set into the ROSDR. The Fetch or Store micro-instruction is then read out to the ROSDR twice. When the Fetch or Store micro-instruction is read out for the second time, no useful function is performed. This is called a dummy cycle.

Three different modes of operation are possible depending upon the sequence of the micro-instructions:

a. If a Fetch or Store micro-instruction is immediately followed by another Fetch or Store micro-instruction, two dummy cycles are required before the core storage is available again.
b. If a Fetch or Store micro-instruction is followed by a micro-instruction other than Fetch or Store, only one dummy cycle is required.
c. The dummy cycle after a Fetch or Store can be omitted by using a short micro-instruction for a Fetch or Store. This, however, must be programmed by the designer of the micro-program, since the short instruction can be used only on one condition: the micro-instruction, which follows the short instruction must not address DR-U and L. One exception to this restriction is that if the short instruction is a Store, data may be read out of DR-U and L.

Summary

If a Fetch or Store micro-instruction is immediately followed by another Fetch or Store, the second F or S can not be executed until the first F or S is finished. Therefore two 0.6 usec dummy cycles are required. During the first dummy cycle, the first F or S is read out from the ROS to the ROSDR a second time but it performs no useful function. In the first dummy cycle, the ROAR Advance is up again and this causes the second F or S micro-instruction to be read out in the second dummy cycle. The second F or S is not executed until the first one is completed.

RING A AND RING C

- ROAR Advance is controlled by the A and C Ring.
- Each Clock A reset-sets Ring A1 Latch.
- Each Clock C reset-sets Ring C1 latch if the Ring A1 latch is on.
- Both Ring A1 and Ring C1 latches are on (as long as no Fetch or Store instruction is called for by the micro-program).
- When a Fetch or Store is decoded, the core storage cycle is started and the ROAR Advance is suppressed.
- The suppression of ROAR Advance causes the last micro-instruction to be read out again. (Dummy cycle)

Description (Figure 2-38)

The Clock A signal which follows the F or S resets the Ring A1 latch and turns on the Ring A2 latch. Refer to figure 2-38. The following Clock C resets Ring C1 latch and turns on Ring C2 latch in a similar manner. The Ring C2 latch allows the ROAR to advance and, with 1 cycle delay, the next micro-instruction is read out. This micro-instruction is executed immediately unless it is another F or S. If it is another F or S, the ROAR Advance is suppressed again, thus causing a second dummy cycle. Clock A of the second cycle after the first F or S resets the Ring A2 latch and turns on the Ring A3 latch. Clock C resets the Ring C2 latch and turns on the Ring C3 latch.

In the second dummy cycle the second F or S is read out (The first F or S was read out during the first dummy cycle) and starts the Core Storage timing.

Clock A of the third cycle after the first F or S resets the Ring A3 latch and turns on the Ring A1 latch. Clock C resets the Ring C3 latch and turns on the Ring C1 latch. With Ring Latch A1 and C1 on, the Ring is back to its normal status.

READ ONLY STORAGE DATA REGISTER (ROSDR)

- **ROSDR is a 22 position latch register capable of containing one complete micro instruction.** (Figure 2-39).

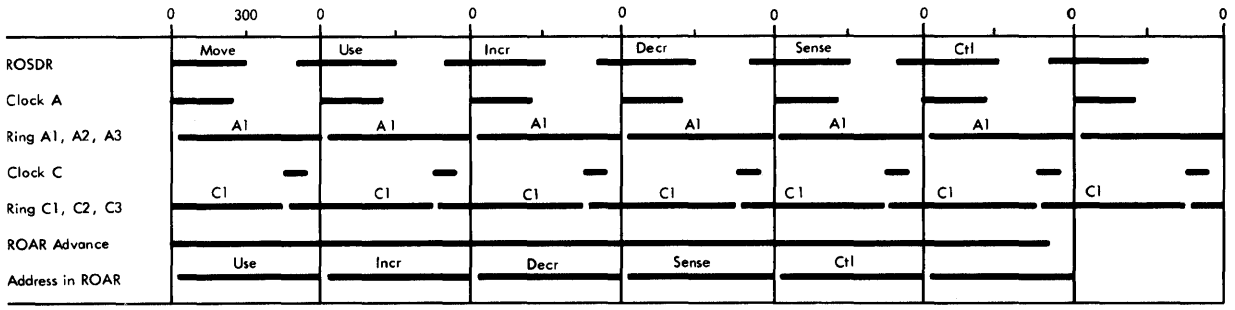


Figure 2-34 ROAR Advance by Move, Use, Incr., Decr, Sense Sequence

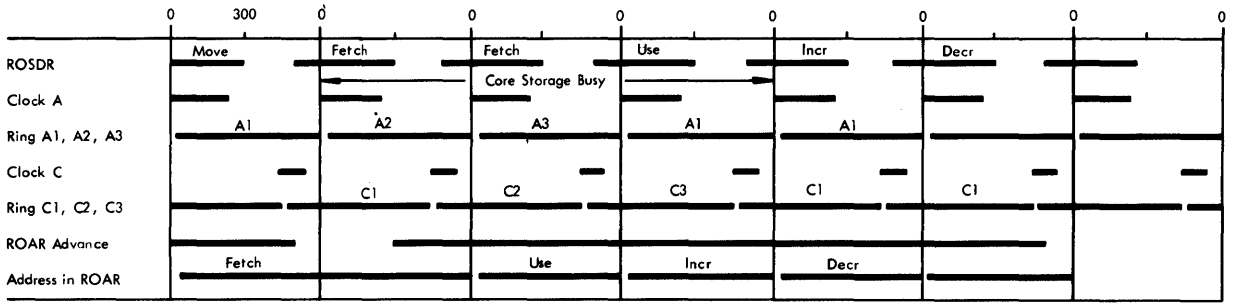


Figure 2-35 ROAR Advance by Move, Fetch, Use, Incr, Decr Sequence

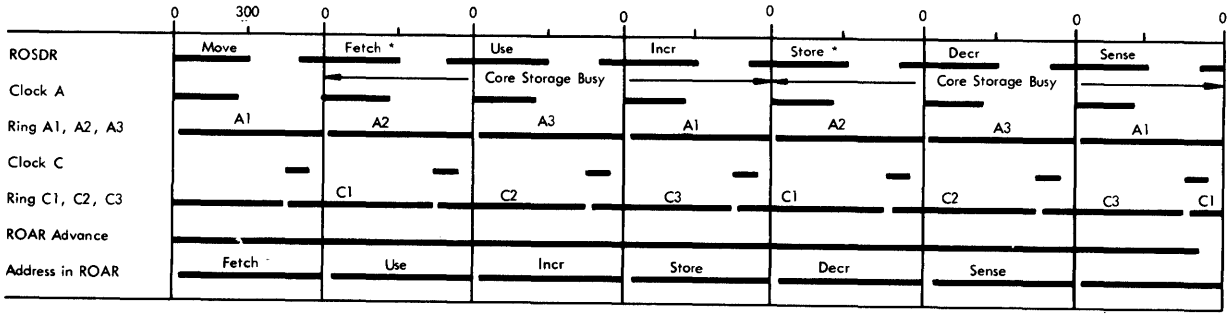


Figure 2-36 ROAR Advance b y Move, Fetch*, Use, Incr, Store* Decr, Sense, Sequence

* (Fetch and Store with Short Cycle)

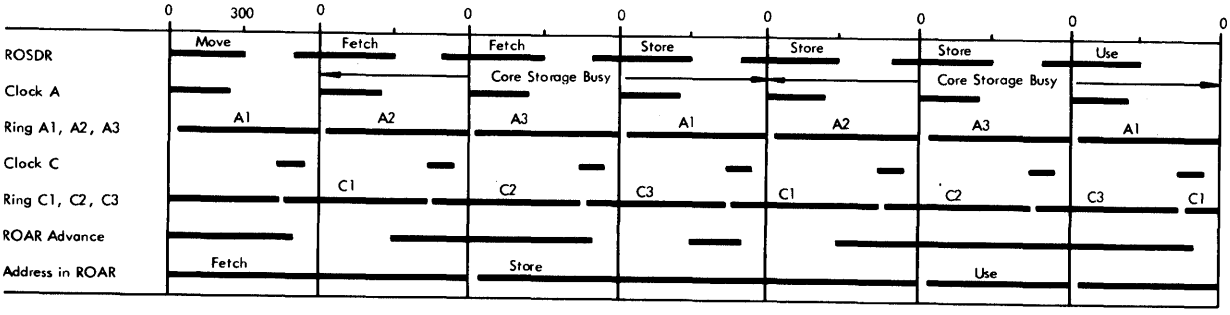


Figure 2-37 ROAR Advance by Move, Fetch, Store, Use Sequence

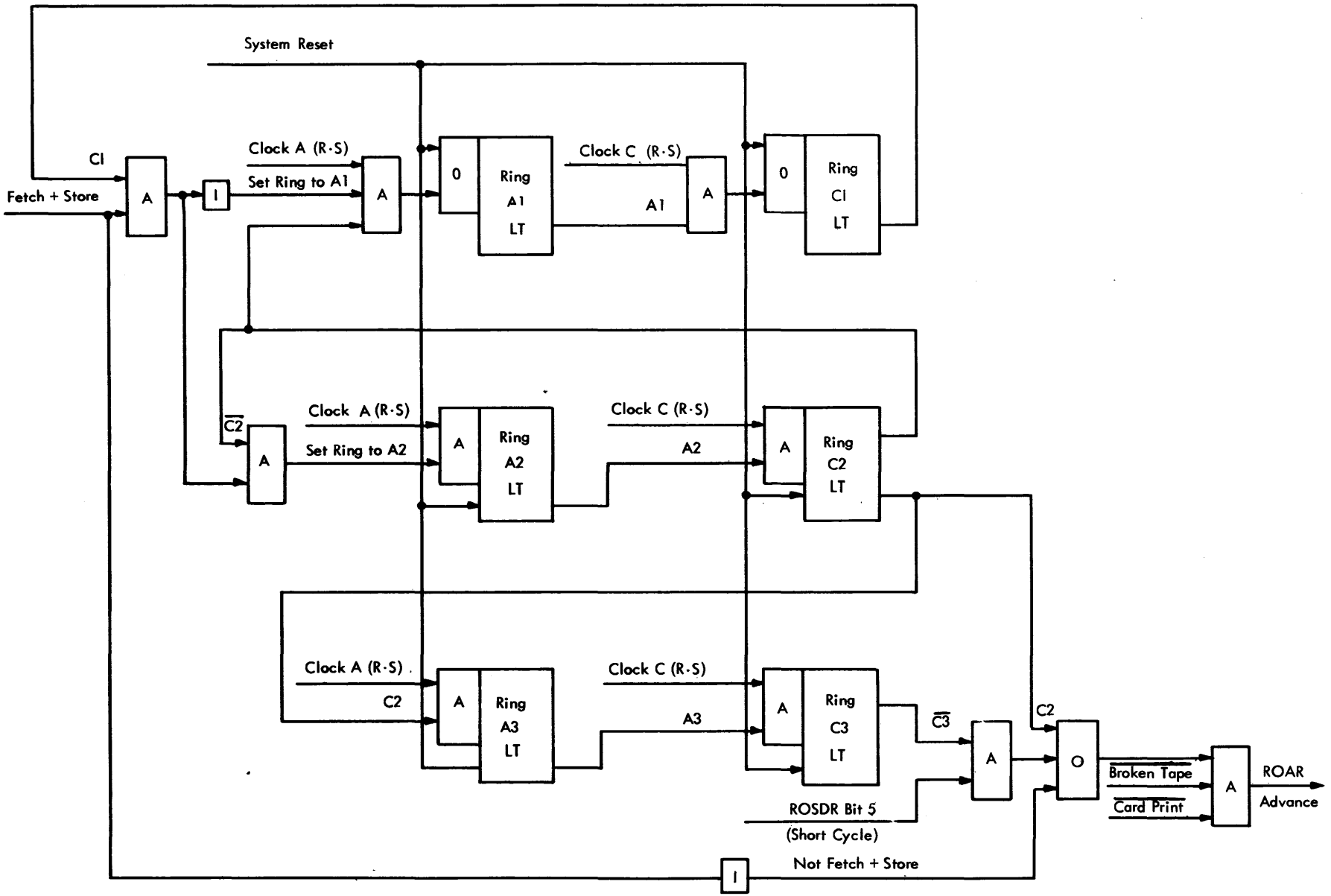


Figure 2-38 A and C Ring

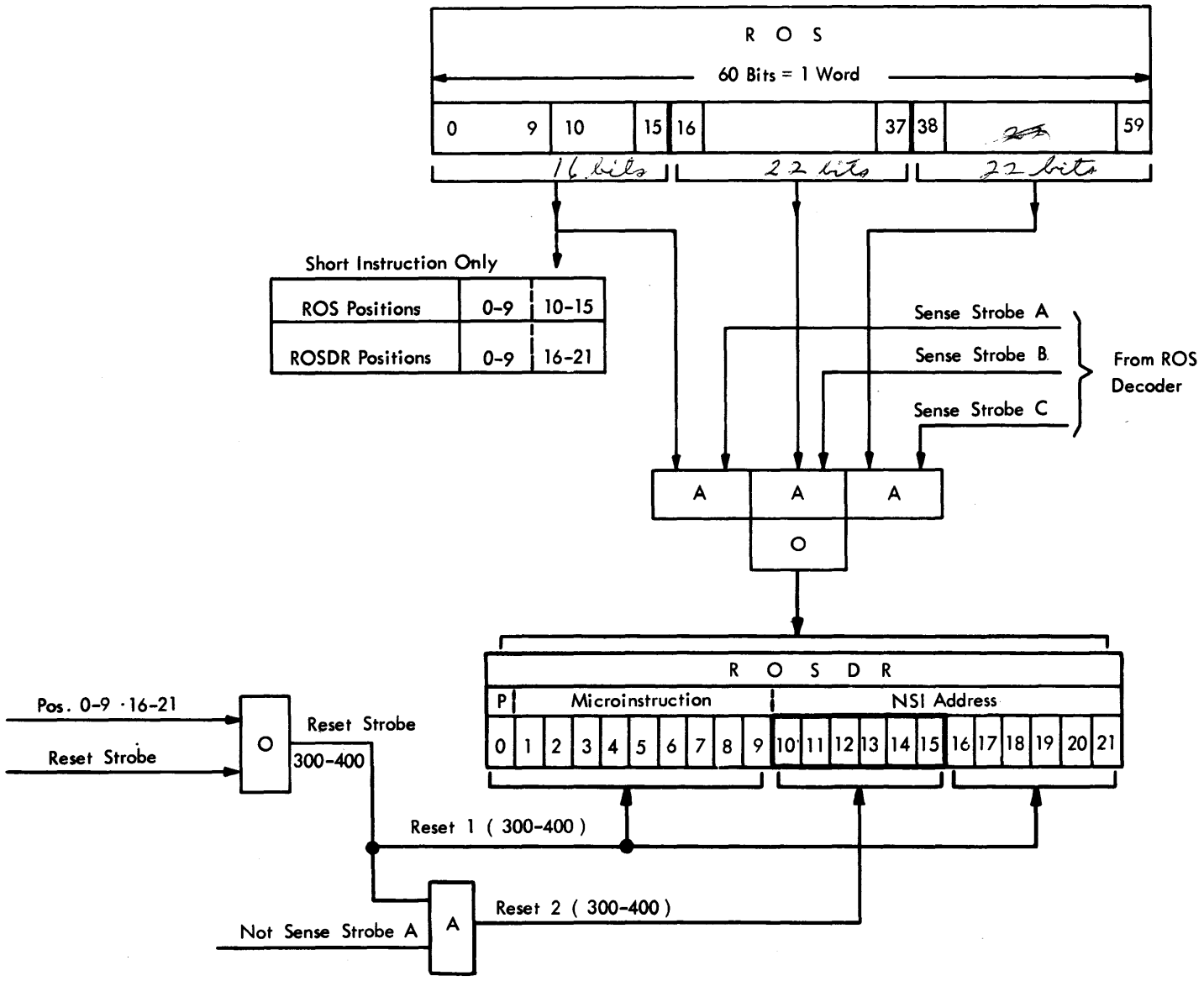


Figure 2-39 ROSDR

- ROSDR receives the micro instruction from the ROS.
- Position 0 contains a parity bit which is checked with the other 21 bits by the ROS Check circuits.
- Positions 1-9 contain the bits which control the execution of the micro instruction.
- Positions 6-9 (N bits) contain immediate data for MX*N, F/SN and F/S PN micro instructions.
- Positions 10-21 contain the Next Sequential Instruction (NSI) address which is set into the Read Only Address Register (ROAR).

READ ONLY ADDRESS REGISTER (ROAR)

- Consists of a 13 bit latch register for containing the address to select any micro instruction in the TROS.
- The Next Sequential address (NSI), in the ROSDR position 10-21, is transmitted to the ROAR.
- Position E (Extension) is turned on if the TROS address is greater than 3072.
- Positions 10-17 of the ROSDR are transmitted to the corresponding ROAR positions.
- Positions 18-21 of the ROSDR are transmitted to the corresponding ROAR position unchanged as long as no USE, INCR, or DECR operation occurs.
- Positions 18-21 of the ROAR can contain a mix of the corresponding ROSDR position and Bus bit (8, 4, 2, 1) depending upon the mask of a USE micro instruction which can alter the input to the ROAR.
- Position 21 is turned on when an INCR or DECR micro instruction causes a carry to occur in the Modifier.
- All ROAR positions except E are turned ON (address 3 F 3 F) by the Start reset signal or by the Card Print optional feature.
- CE switches provide for entering any combination of bits into the ROAR.

TROS TAPE SELECTION EXAMPLE (TROS ADDRESSING)

Figure 2-41 shows TROS Tape No. 019. The outer Word of this tape was addressed in the previous ROS-cycle. Although the whole Word was addressed, only instruction C was set into ROSDR. (It is always only one of the three instructions that is set into ROSDR). The actual micro-instruction is in ROSDR positions 0-9. This instruction is decoded in the ROS Op decoder. This decoder contains gate decoders which have access to the Bus and to the Data Registers.

ROSDR positions 10-21 contain the address of the next micro-instruction. Note that this address is actually short by 1 bit, since it can address only addresses up to Module 3. To facilitate addressing of all 8 Modules an Extension bit is needed. The ROAR bit E is therefore turned on when ROSDR bits 16 and 17 are both zero. Bit E is turned off when bit 17 is turned on while bit 16 is off. Bits 16 and 17 serve in a double function since they also select the instruction of the next TROS Word (instr. C in this case). The bits in ROAR address the next Word by activating ROS gate decoders (10 and 0 in this example) and ROS drive decoders (40 and 5). The gate and drive decoders select the final gate and the final driver. It is the final gate and driver that addresses the next Word. Due to the sense strobe that was generated by ROAR bits 16 and 17 the third instruction of this next Word will be set into ROSDR. The selection of the inner or outer Word depends on ROAR bit 13 . When 13 is off (zero) the inner Word is selected and vice versa. The next instruction (NSI) can be determined quicker by using the table (Figure 2-42). (Also read the explanations on CAS-sheet representation (appendix) for further operating examples).

Broken TROS Tape. "Tape" is the expression for the TROS mylar tape in which a 60 bit word is punched. Broken tape is detected, if any micro instruction puts no bits into ROSDR positions 1 through 5.

GATE DECODERS

- ROSDR positions 7 through 9 contain the bits which activate the lines to gate a DR to the bus. ROSDR positions 7-9 are labeled gate.
- Two gate Decoders work in parallel (Gate Decoder 1 and 2) to decode 8 different gates labeled 0 through 7.
- Gates 0 through 7 are related to gating DR-E, S, T, R, U, L, P, I to the bus, i.e., DR-E/, DR-S/1, DR-T/2, DR-R/3, etc., to DR-I/7.

ADDRESS HOLD REGISTER (AHR)

- Consists of 4 latches bit P, 4, 2, and 1 and holds (retains) the address of any DR which is reset-

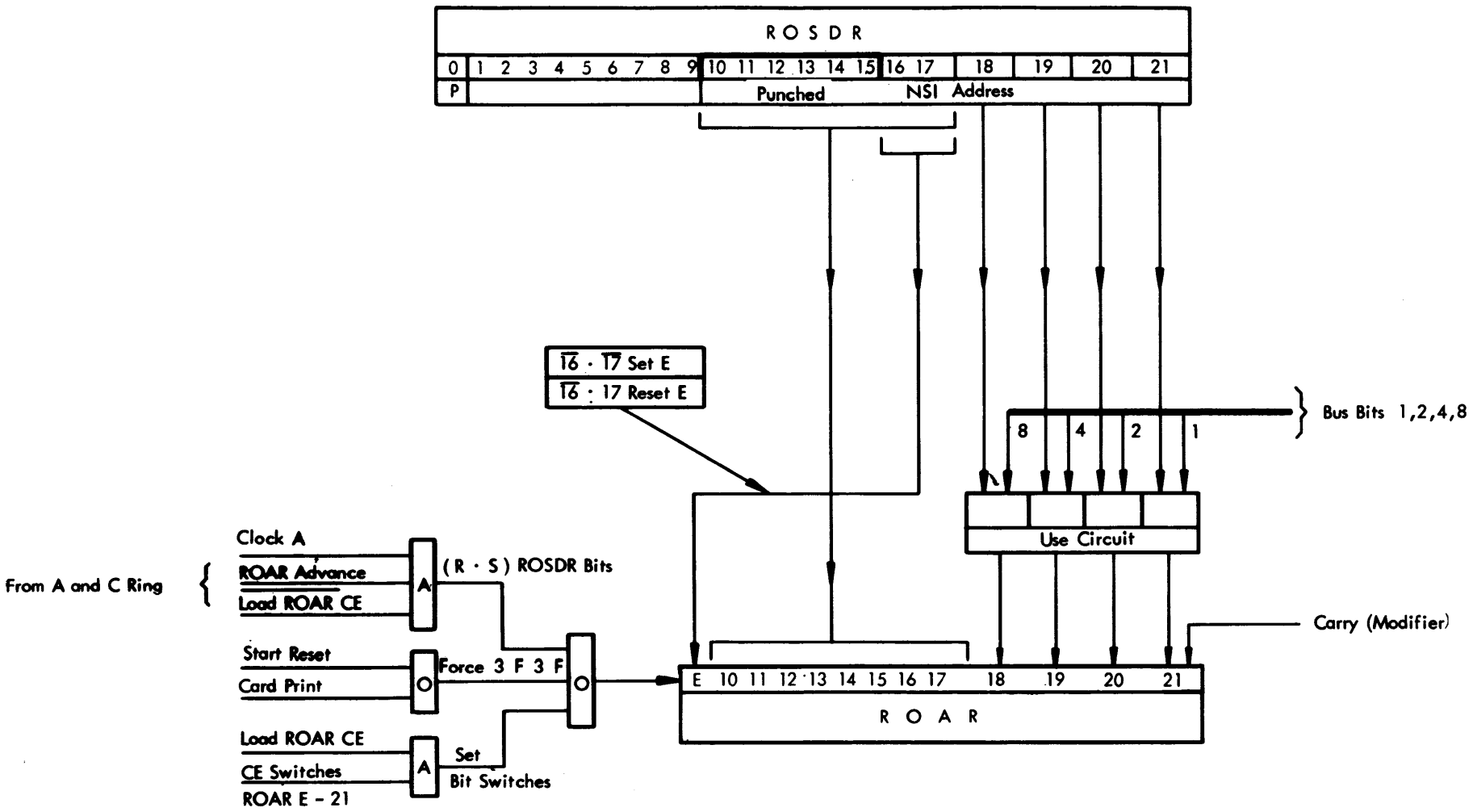


Figure 2-40 ROAR

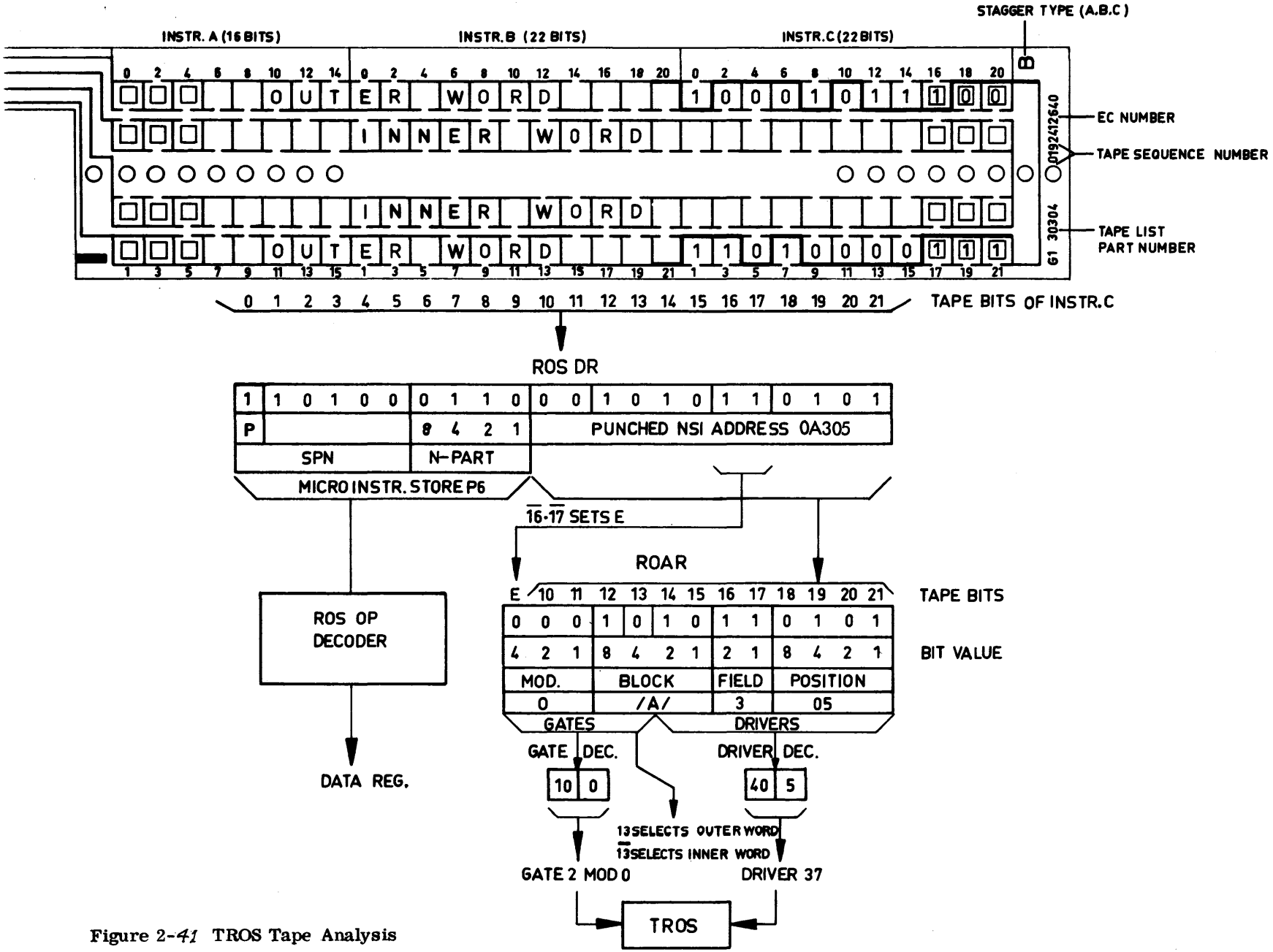
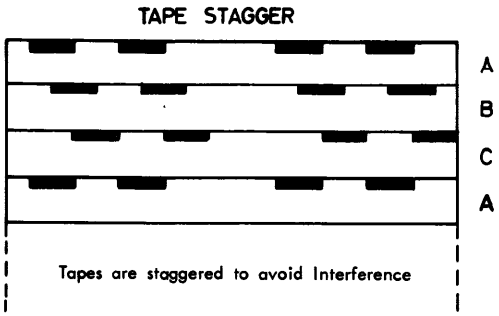
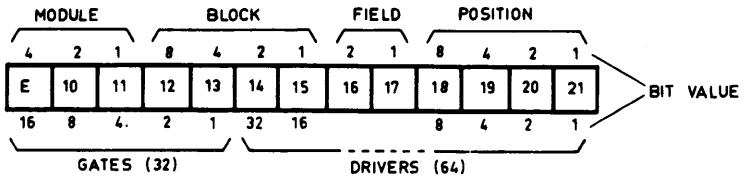
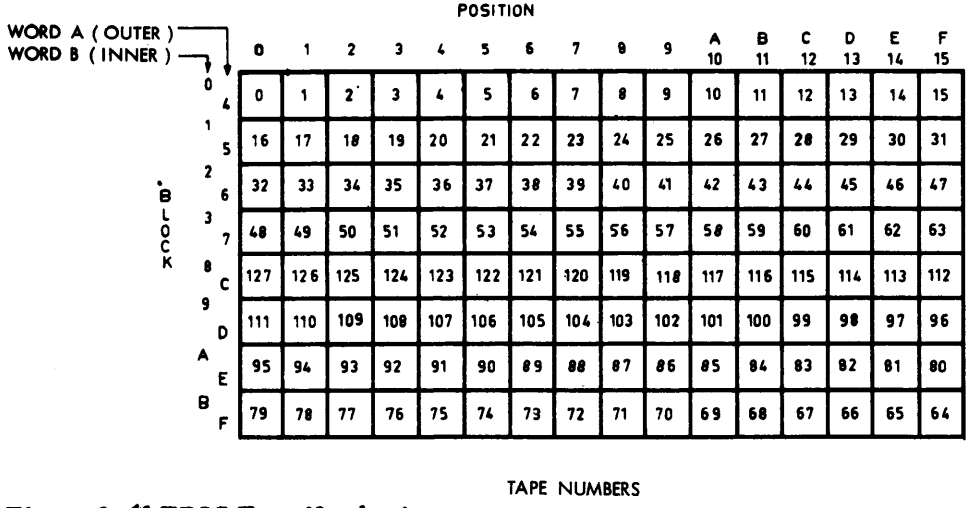


Figure 2-*41* TROS Tape Analysis







LIST OF TAPES SHOWN IN ORDER OF STACKING

TAPE STAGGER A	TAPE STAGGER B	TAPE STAGGER C
	64	65
66	67	68
69	70	71
72	73	74
75	76	77
78	79	80
81	82	83
84	85	86
87	88	89
90	91	92
93	94	95
96	97	98
99	100	101
102	103	104
105	106	107
108	109	110
111	112	113
114	115	116
117	118	119
120	121	122
123	124	125
126	127	
RESISTIVE TAPE 14-1425		
63		
60	61	62
57	58	59
54	55	56
51	52	53
48	49	50
45	46	47
42	43	44
39	40	41
36	37	38
33	34	35
30	31	32
27	28	29
24	25	26
21	22	23
18	19	20
15	16	17
12	13	14
9	10	11
6	7	8
3	4	5
0	1	2

Figure 2-42. TROS Tape Numbering

set by an MX*X, MX*N, INCR X or DECR X micro instruction.

- The data register address must be retained (temporarily stored) because the ROSDR is reset before the data register can be set.

Circuit Description (Figure 2-43)

The micro-instructions MX*X, MX*N, INC X, and DECR X all address a selected data register to which data are to be transmitted. The address of the data register comes from the ROSDR. The ROSDR is reset at time 300-400 by its reset strobe. The data register, however, is reset/set by clock C at time 400-550. This means that the address from the ROSDR would actually be lost before it could address the selected DR. Therefore the address is held in the Address Hold Register which is set/reset at time 0-300 by clock A (Figure 2-43). The address retained in the AHR is used only when a MX*X, MX*N, NCR X, or DEC X micro-instruction is programmed.

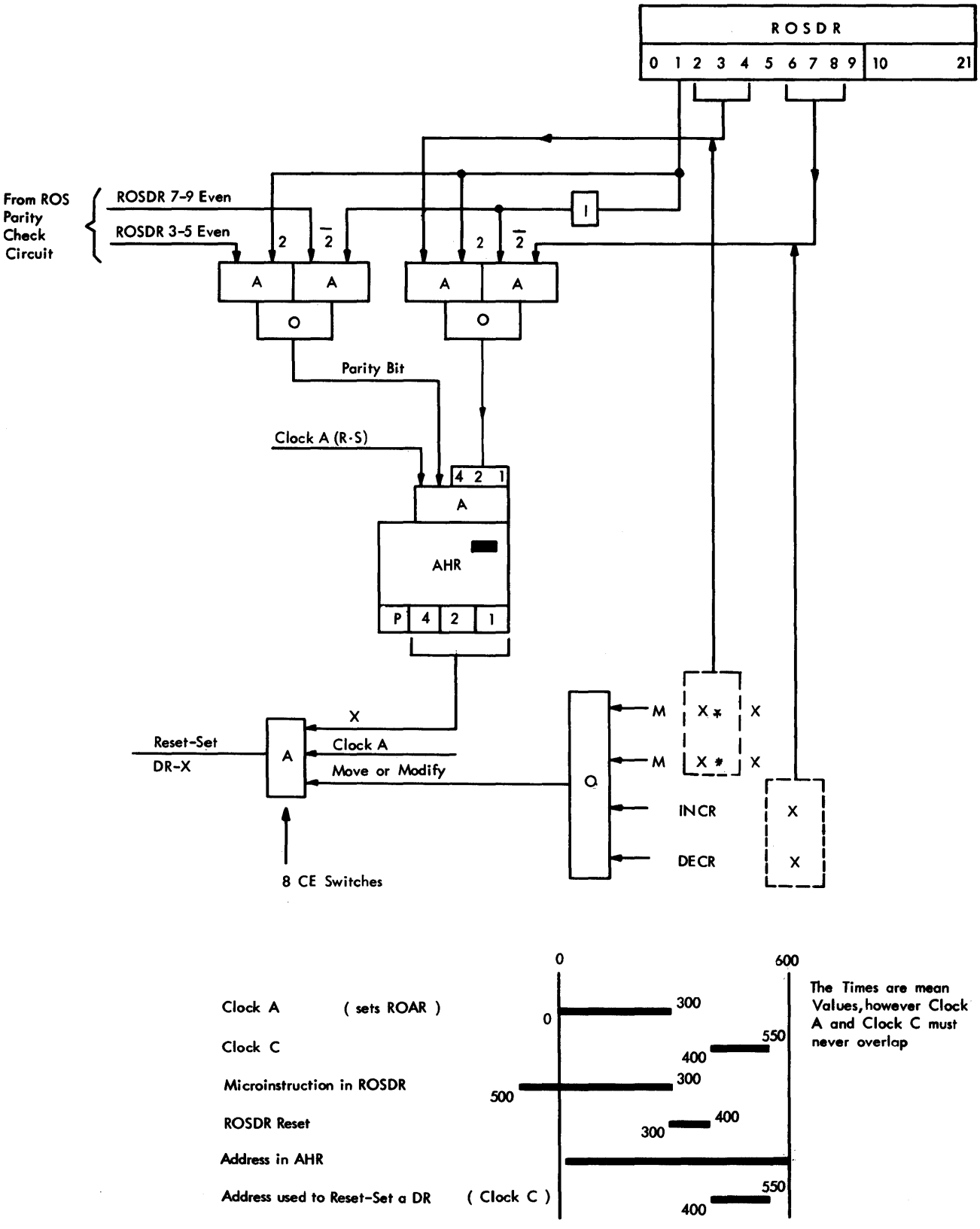


Figure 2-43 AHR Control and Timing

When the Start key is pressed, the micro-program (Which is stored in the Read Only Storage) is started. The Micro-program carries out a number of general operations which handle all individual CPU operations. The first one of these general operations is the manual routine. Refer to Figure 3-1.

In the manual routine all Data Switches on the Customer Panel (which may be set up manually) are sensed and the commands which these switches represent are executed. For example:

> If the Mode switch is in the Register Display position (Instead of the normal Process position) the CPU does nothing more than display the desired register.

The second general operation is the Interrupt Routine. This Routine is executed only when necessary, that is, when an I/O device has signalled that it is finished with its current operation. The interrupt routine itself is a branch to a main program-sub-routine made by the customer.

Next, the Op Code of the first instruction of the customers program is read out of core storage. This Op Code is tested for validity and its format is decoded.

Once the format is known, the Op Code is further decoded in the I phase, which is common to all Op Codes, to determine exactly what is to be done with the data of the instruction.

Each Op Code has its own E-phase. During the E-phase, the actual data processing is executed, for example Add, Subtract, Edit and so on.

After the E phase, the micro-program returns to the manual routine to start the same sequence of operations again. Thus, the CPU processes data by going through the Manual Routine, the Interrupt, the I-phase, the E-phase and back to the Manual Routine.

Among the Op Codes in the SI and SS Format are some that concern the I/O devices. When the decoding in the I-phase has revealed that the Op Code affects an I/O device, a common E-phase is entered.

In the Common E-phase, it is decided whether a Test I/O and Branch or a Control I/O instruction is present. The Op Codes are tested for validity and the Function Specification is decoded. This is necessary because the Op Code for Read Secondary is identical with the Op Code for Punch Primary. Only the Function Specification marks the difference. Some instructions contain a Detailed Function Specification which specifies, for example, exactly which print head must be selected.

After the Common E phase, the Device Address is tested to find out exactly which device is to carry out the instruction.

Each device has its own Special E-phase. In the special E-phase, the necessary Execute latch is set. This starts the drive motor of the addressed device. In this manner, the mechanism of the I/O device is already in motion when the Service Phase begins.

At the end of the Special E-phase, the device makes a request for Service. These Service Requests are recognized by the CPU because throughout the micro-program, in fact after every 72 micro instructions, one instruction tests to determine whether there is Any I/O Request. If there is, the micro-program first handles the Service Phase before it continues with its normal tasks (I-phase, Manual Routine or whatever it may be at the moment).

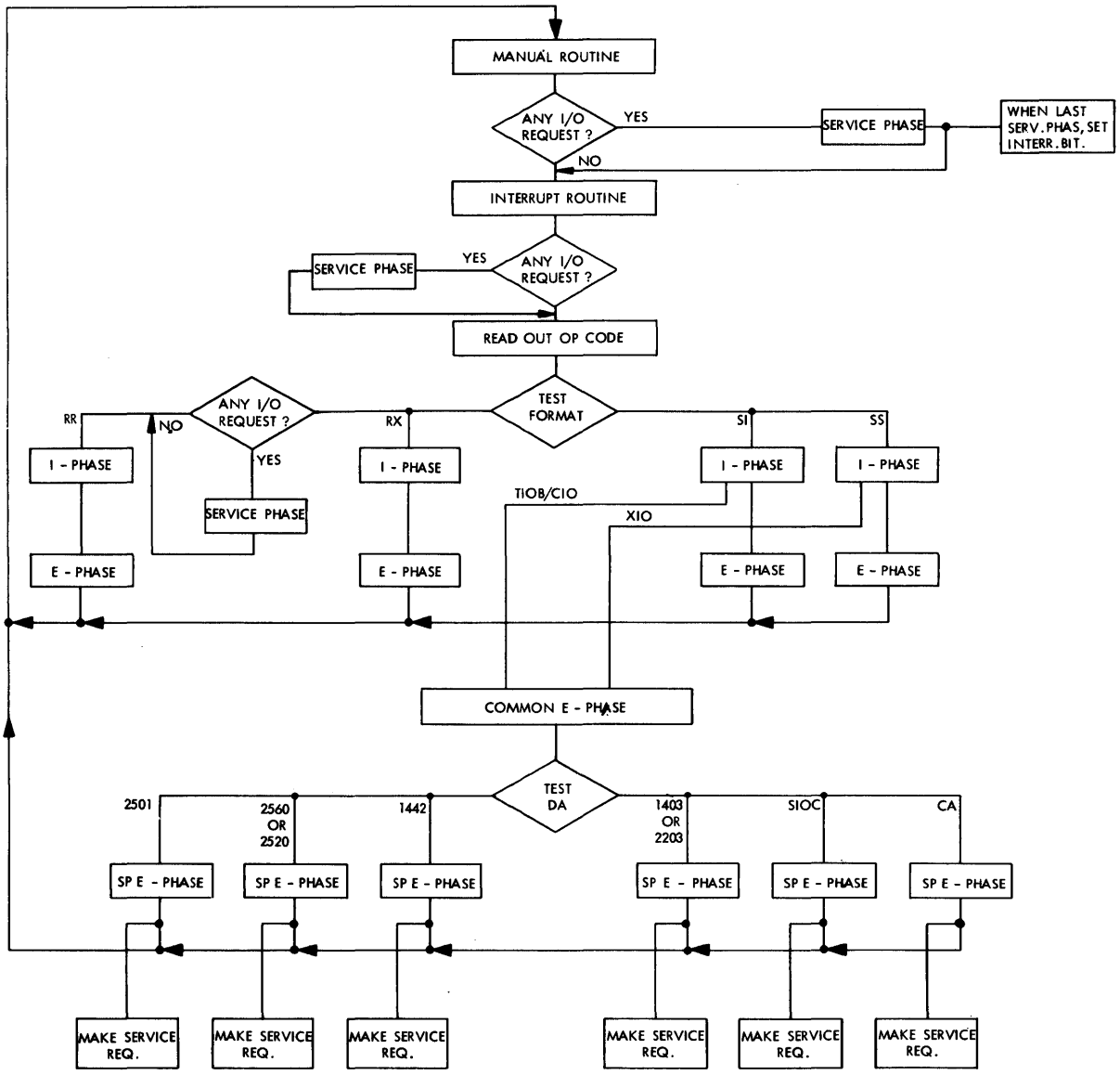
In the Service Phase only one card column (for example) of perhaps 60 programmed ones is read into the read in area of the core storage. So, little by little (Serv. Phase by Serv. Phase) all programmed columns are read in. This is the time sharing principle of the Model 20. When the last column is read in, the interrupt bit is set into Auxiliary Storage. When the CPU comes to the interrupt routine it executes it.

MICRO-PROGRAM OPERATIONS

MICRO-INSTRUCTIONS

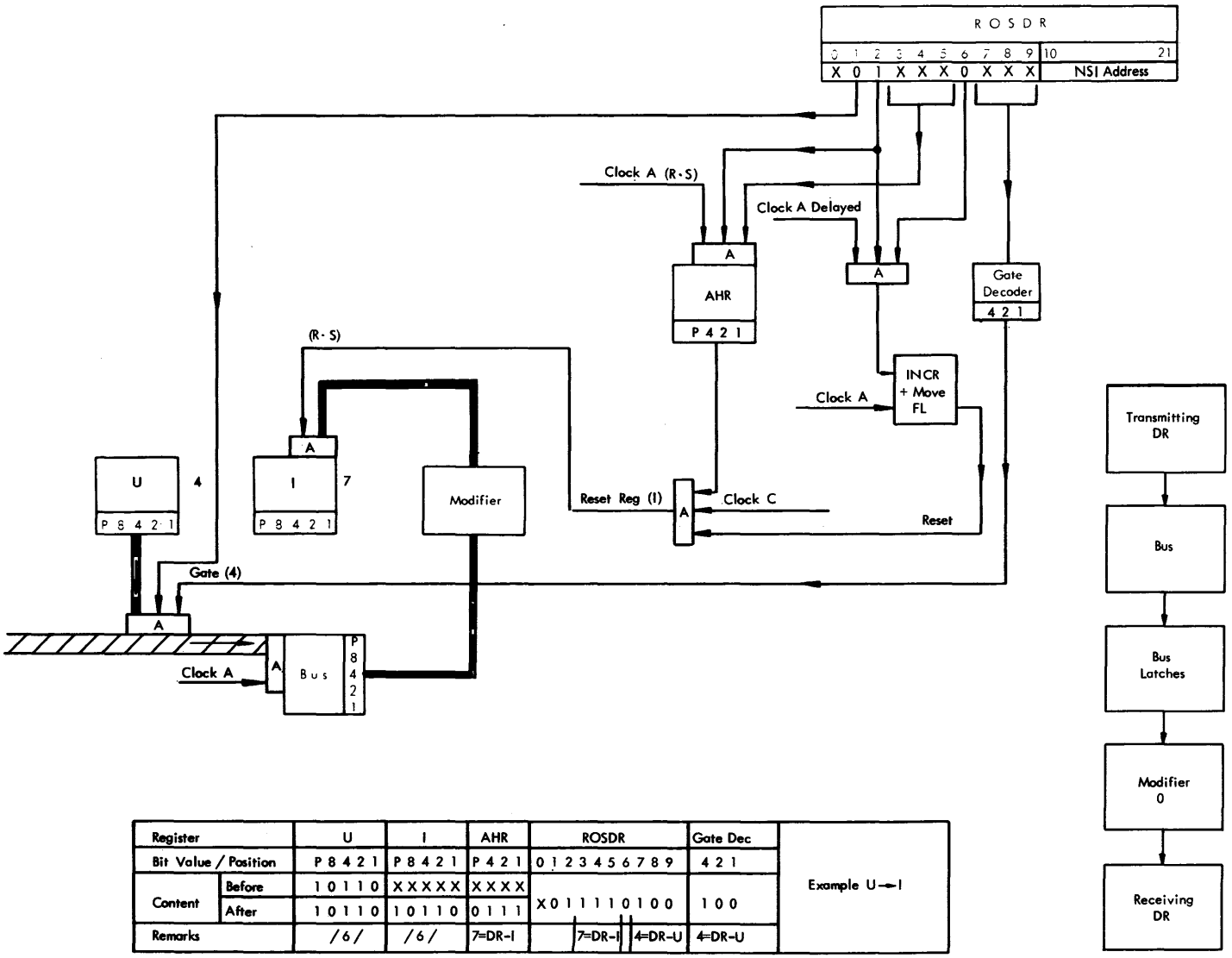
MX*X (Figure 3-1a)

- Objective: Transmit the content of a selected Data Register (DR) designated by X into another selected DR designated by (X*)
- "M" is the micro-instruction Op code for MOVE
- "X*" is the address of the selected DR which receives the data
- "X" is the address of the selected DR which transmits the data



(Any I/O Requests occur every 42μsec. anywhere in the Micro Program. Only 3 Requests are shown here.)

Figure 3-1 General CPU Operation



Register		U	I	AHR	ROSDR	Gate Dec	
Bit Value / Position		P 8 4 2 1	P 8 4 2 1	P 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1	Example U→I
Content	Before	1 0 1 1 0	X X X X X	X X X X	X 0 1 1 1 1 0 1 0 0	1 0 0	
	After	1 0 1 1 0	1 0 1 1 0	0 1 1 1			
Remarks		/6/	/6/	7=DR-I	7=DR-I 4=DR-U	4=DR-U	

Figure 3-1a MX-X Data Flow

- Figure 3-1 is a detailed diagram of the data paths and controls and includes an example to move the content of DR-U (/6/) to DR-I.

Circuit Objectives

ROSDR Pos. 7, 8, 9 X X X	are variable depending upon the address of the transmitting DR. This address is decoded in the Gate Decoder.
ROSDR Pos. 1 (0)	is a zero (0). The 0 is ANDed with the output of the Gate Decoder to gate the transmitting DR to the BUS.
ROSDR Pos, 3, 4, 5 X X X	are variable depending upon the address of the receiving DR.
ROSDR Pos. 2 (1)	is a one (1). The 1 is ANDed with the receiving DR address to set the AHR (Address Hold Register).
ROSDR Pos. 2,6 (1 0)	2 is a one (1) and 6 is a zero (0). These two conditions ANDed set the Incr or Move latch. The "Reset" signal from the Incr or Move latch is ANDed with the DR address in the AHR register to reset and set the receiving DR.

MX*N (Figure 3-2)

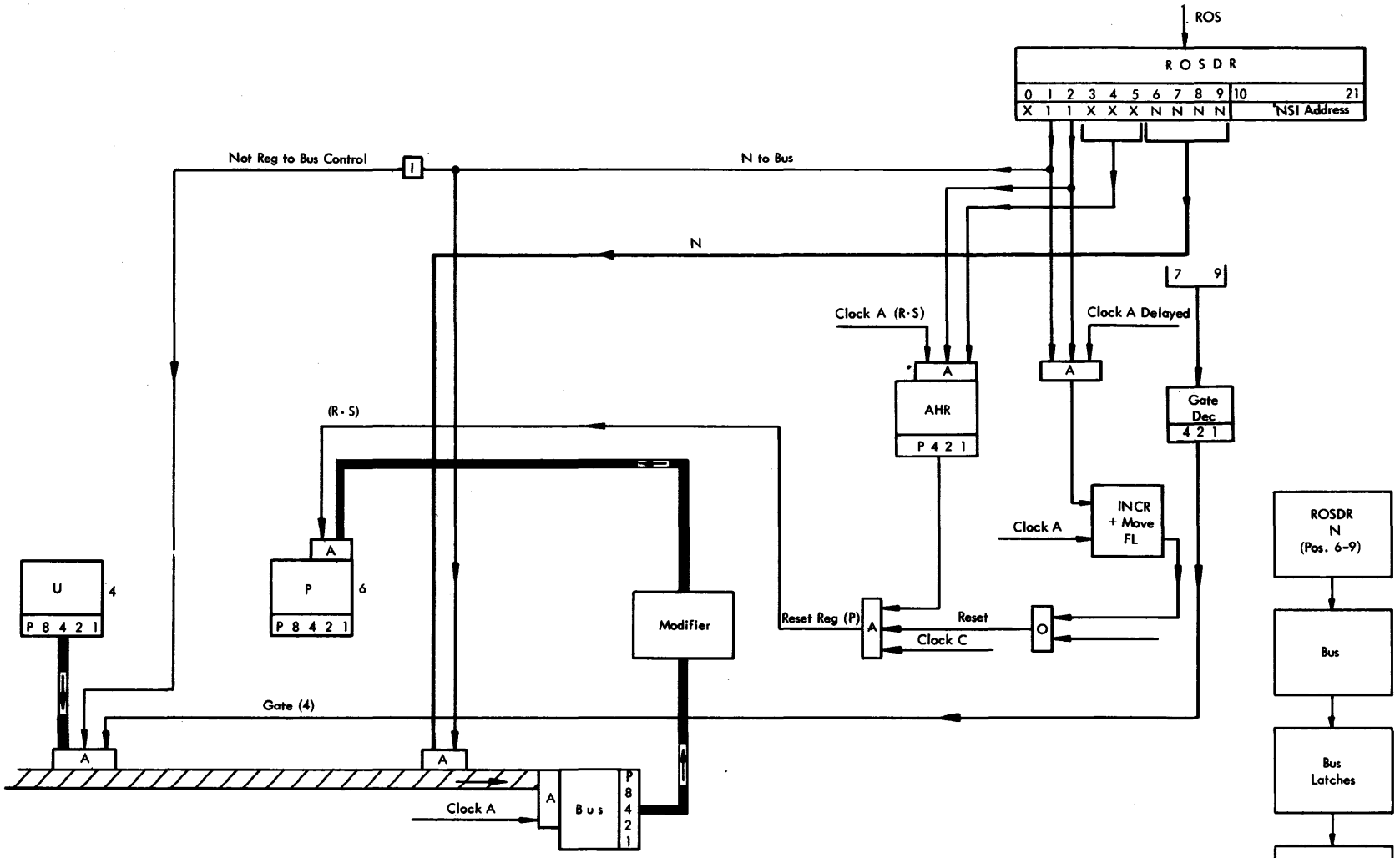
- Objective: Transmit the "N" part of the micro-instruction (a binary value between 0 and 15) into a selected data register (X*).
- "M" is the micro instruction OP code for MOVE
- "X*" is the address of the receiving data register
- "N" designates positions, 6, 7, 8 and 9 of the ROSDR. Position 6, 7, 8, and 9 of ROSDR contain the data from position 6, 7, 8 and 9 of the move micro-instruction.
- Figure 3-2 is a detailed diagram of the data paths and controls and includes an example to move the "N" part (11) of this micro-instruction into DR-P.

Circuit Objectives

ROSDR Pos. 6, 7, 8, 9 X X X X	"N" part is variable depending upon the binary value (between 0 and 15) which is moved.
ROSDR Pos. 1 (1)	is a one (1) which gates the "N" part to the Bus, prevents the GATE DECODER output from gating any DR to the BUS.
ROSDR Pos. 3, 4, 5 X X X	are variable depending upon the address of the receiving DR.
ROSDR Pos. 2 (1)	is a one (1) which gates ROSDR positions 3, 4, 5 (the receiving DR address) into the AHR (Address Hold Register).
ROSDR Pos. 1, 2 (1 1)	are both ones (1) which set the Incr or Move latch. The reset signal from the Incr or Move latch is ANDed with the DR address in AHR to reset and set the receiving DR.

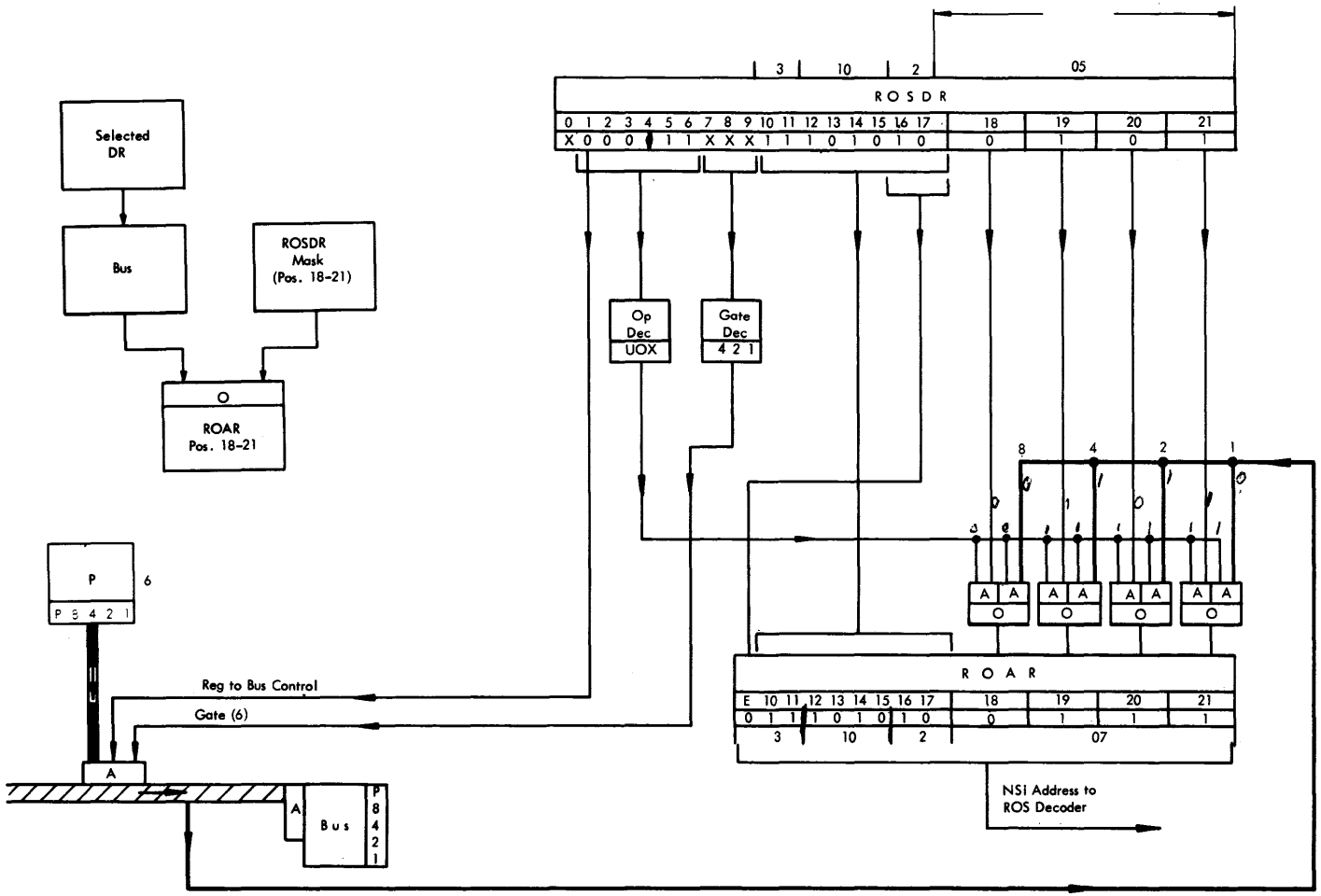
UOX (UX) (Figure 3-3)

- Objective: Use the contents of a selected DR (X), ORing it with the NSI address to determine if a branch in the micro-program is required.
- "U" is the micro-instruction OP code for Use.
- "O" Means OR the MASK (the last four positions of the NSI address, located in position 18, 19, 20, and 21 of the ROSDR) with the contents of a selected DR and gate the result into the last four positions of the ROAR (position 18, 19, 20, 21).
- The 0 is normally not printed, if a UOX is shown.
- "X" is the address of the selected DR.



Register		U	P	AHR	ROSDR			Gate Dec	
Bit Value/ Position		P 8 4 2 1	P 8 4 2 1	P 4 2 1	0 1 2 3 4 5 6 7 8 9			4 2 1	Example 4 → P
Content	Before	X X X X X	X X X X X	X X X X	X 1 1 1 1 0 1 0 1 1			1 0 0	
	After	X X X X X	0 1 0 1 1	1 1 1 0					
Remarks			/11/	6=DR-P		DR-P	N=11	4=DR-U	

Figure 3-2 MX-N Data Flow



Register		P	ROSDR	Gate Dec		
Bit Value / Position		P 8 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1		Example UP 0101
Content	Before	1 0 0 1 1	X 0 0 0 1 1 1 1 1 1 0	1 1 0	Old NSI Address 3 10 2 05	
	After	1 0 0 1 1			New NSI Address 3 10 2 07	
Remarks		3	DR-P	6=DR-P		

Figure 3-3 UOX Data Flow

- Figure 3-3 is a detailed diagram of the data paths and controls and includes an example to generate a new NSI address (punched NSI address is 3 10 2 05; new NSI address is 310207) depending on the content of DR-P (DR-P contains a 3).

Description

This micro-instruction gates the data from the selected DR via the bus through an OR circuit into the ROAR. The Mask is handled in the same manner because the last four bits of the ROSDR are gated through the same OR circuit into the ROAR. When the Mask and the DR bits are ORed, and the result is equal to the mask bit pattern, no branch occurs. When the mask and the DR bits are ORed, and the result is not equal to mask bit pattern, a branch occurs. The result of the ORing is set into ROAR positions 18, 19, 20, and 21. The address in ROAR is the address of the NSI.

Circuit Objectives

ROSDR Pos. 7, 8, 9 X X X	are variable depending upon the address of the selected DR. This address is decoded in the Gate Decoder.
ROSDR Pos. 1 (0)	is a zero (0) which is ANDed with the output of the Gate Decoder to gate the selected DR to the BUS.
ROSDR Pos. 1, 2, 3, 4, 5, 6 (0 0 0 1 1 1)	1 - 3 are zeros (0); 4 - 6 are ones (1). This bit pattern is gated into the Op Decoder (UOX). The output of the Op Decoder is ANDed with the mask bits and also, but separately, with the bus bits. The output of each pair of these ANDs are ORed into the ROAR positions 18 through 21 (Figure M6). When either the mask or the selected DR contains a one (1) bit, the corresponding position in the ROAR is set to a one bit.

UAX (Figure 3-4)

- Objective: Use the contents of a selected DR (X) ANDing it with the NSI address to determine if a branch in the micro-program is required.
- "U" is the micro-instruction OP code for USE.
- "A" means AND the MASK (the four last positions of the NSI address, located in position 18, 19, 20, and 21 of the ROSDR) with the content of a selected DR and gate the result into the four last positions of the ROAR (position 18, 19, 20, and 21).
- "X" is the address of the selected DR.
- Figure 3-4 is a detailed diagram of the data paths and controls and includes an example to generate a new NSI address (punched NSI address 3 10 2 05, new NSI address 3 10 2 01) depending on the contents of DR-P (3).

Description

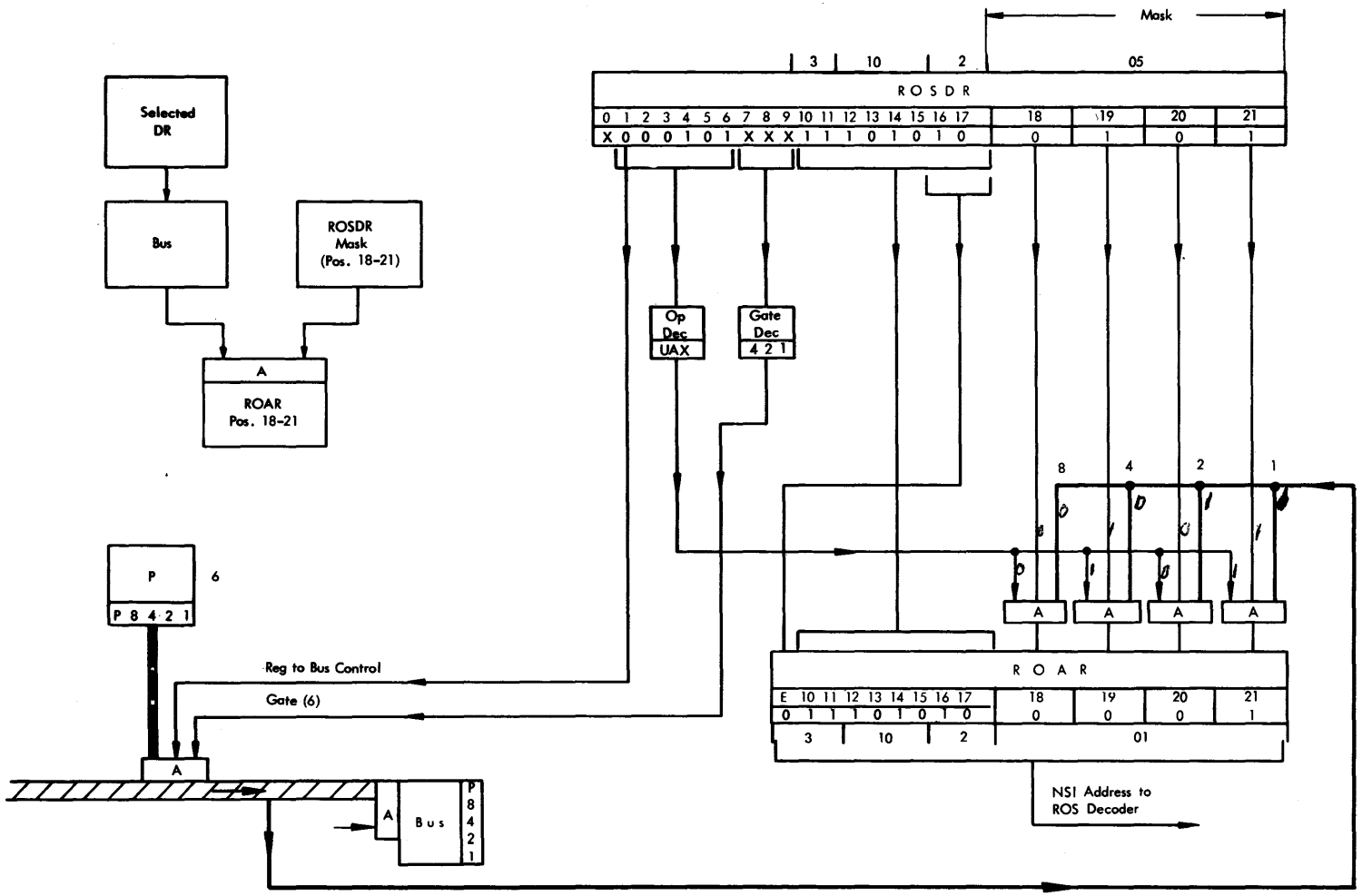
This micro-instruction gates the data from the selected DR, via the bus to an AND circuit. The Mask is handled in the same manner because the last four bits of the ROSDR are gated to the same AND circuit. The output of this AND circuit is set into the ROAR.

When the Mask and the DR bits are ANDed and the result is equal to the mask bit pattern, no branch occurs.

When the mask and the DR bits are ANDed and the result is unequal to the mask bit pattern, a branch occurs. The result of the ANDing is set into ROAR position 18, 19, 20, and 21. The address in ROAR is the address of the NSI.

Circuit Objectives

ROSDR Pos. 7, 8, 9 (X X X)	are variable depending upon the address of the selected DR. This address is decoded in the Gate Decoder.
ROSDR Pos. 1 (0)	is a zero (0) which is ANDed with the output of the Gate Decoder to gate the selected DR to the bus.



Register		P	ROSDR	Gate Dec		
Bit Value / Position		P 8 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1		
Content	Before	1 0 0 1 1	X 0 0 1 0 1 1 1 1 0	1 1 0	Old NSI Address 3 10 2 05	Example UAP 0101
	After	1 0 0 1 1			New NSI Address 3 10 2 01	
Remarks		3	DR-P	6=DR-P		

Figure 3-4 UAX Data Flow

ROSDR Pos. 1,2,3,4,5,6 (0 0 0 1 0 1) 1, 2, 3, and 5 are zeros (0); 4 and 6 are ones (1). This bit pattern is gated into the Op Decoder (UAX). The output of the Op Decoder is ANDed with the Mask bits of the ROSDR and the content of the bus (selected DR) to set ROAR positions 18-21.

A one (1) bit is set into a ROAR position only when both the MASK and the BUS have a one (1) bit in the same corresponding position.

UX_1X_2 (Figure 3-5)

- Objective: USE the content of a selected DR (X_2) and EXCLUSIVE OR it with the NSI address to determine if a branch in the micro-program is required.
- "U" is the micro-instruction OP code for USE.
- "X_1" means to Exclusive OR the mask (the last four positions of the NSI address, located in position 18, 19, 20, and 21 of the ROSDR) with the content of a selected DR and gate the result into the four last positions of the ROAR (position 18, 19, 20, and 21).
- "X_2" is the address of the selected DR.
- Figure 3-5 is a detailed diagram of the data paths and controls and includes an example to generate a new NSI address (punched NSI address 3 10 2 05, new NSI address 3 10 2 06) depending on the contents of DR-P (3).

Description

This micro-instruction gates the data from the selected DR via the bus through an Exclusive OR circuit. The mask is handled in the same manner because the last four bits of the ROSDR are gated through the same Exclusive OR circuit into the ROAR.

When the mask and the DR bits are Exclusive ORed and the result is equal to the mask bit pattern, no branch occurs.

When the mask and the DR bits are EXCLUSIVE ORed and the result is not equal to the mask bit pattern, a branch occurs.

The result of the Exclusive ORing is set into ROAR position 18, 19, 20 and 21. The address in the ROAR is the address of the NSI.

Circuit Objectives

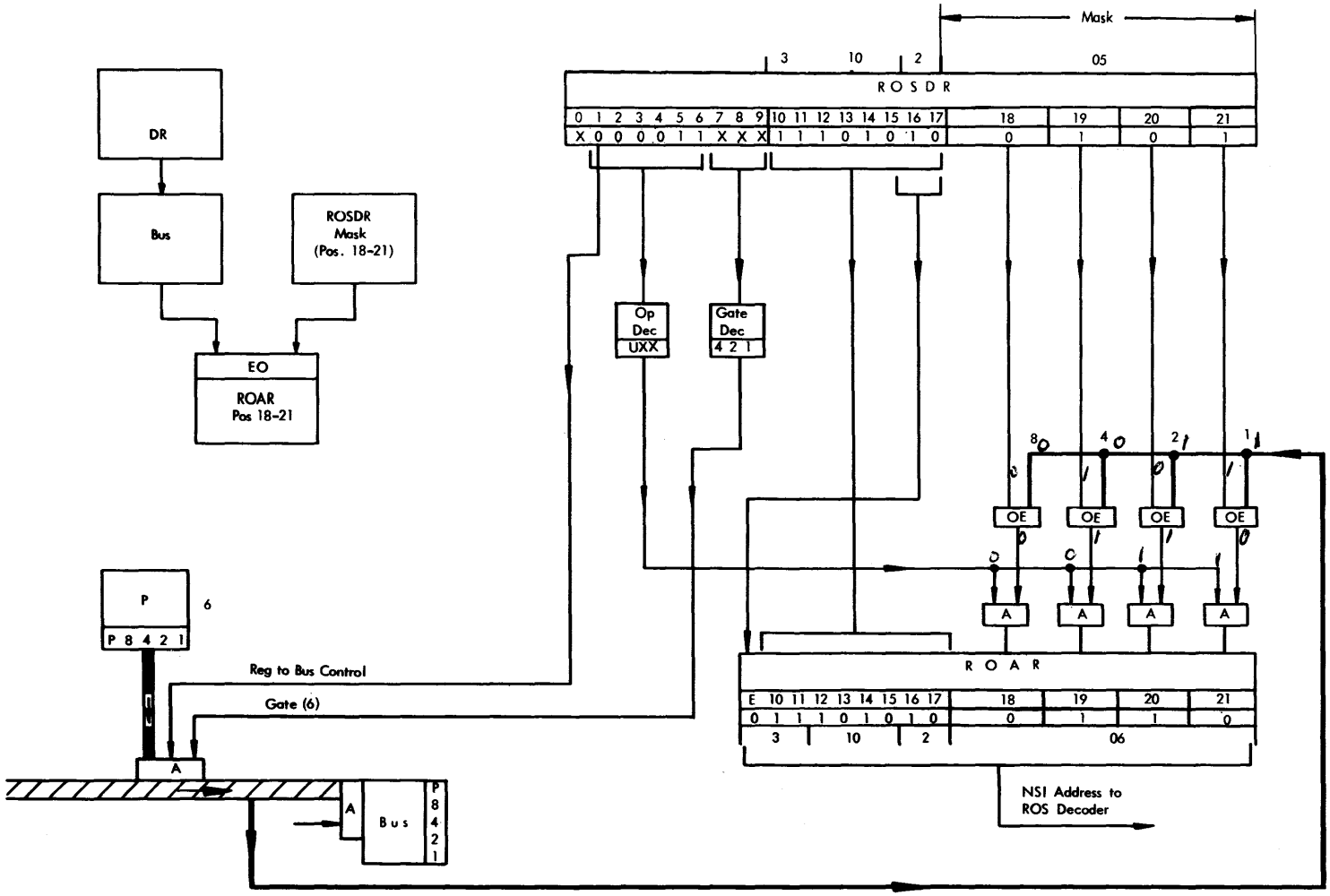
ROSDR Pos. 7, 8, 9 (X X X) are variable depending upon the address of the selected DR. This address is decoded in the Gate Decoder.

ROSDR Pos. 1 (0) is a zero (0). The 0 is ANDed with the output of the Gate Decoder to gate the selected DR to the bus.

ROSDR Pos. 1,2,3,4,5,6 (0 0 0 0 1 1) 1 through 4 are zeros (0); 5 and 6 are ones (1). This bit pattern is gated into the Op Decoder (UXX) The output of the Op Decoder is ANDed with the output of an Exclusive OR circuit to set the ROAR positions 18-21. The Exclusive OR circuit has an output if either the mask or the corresponding bus position has a one (1) bit.

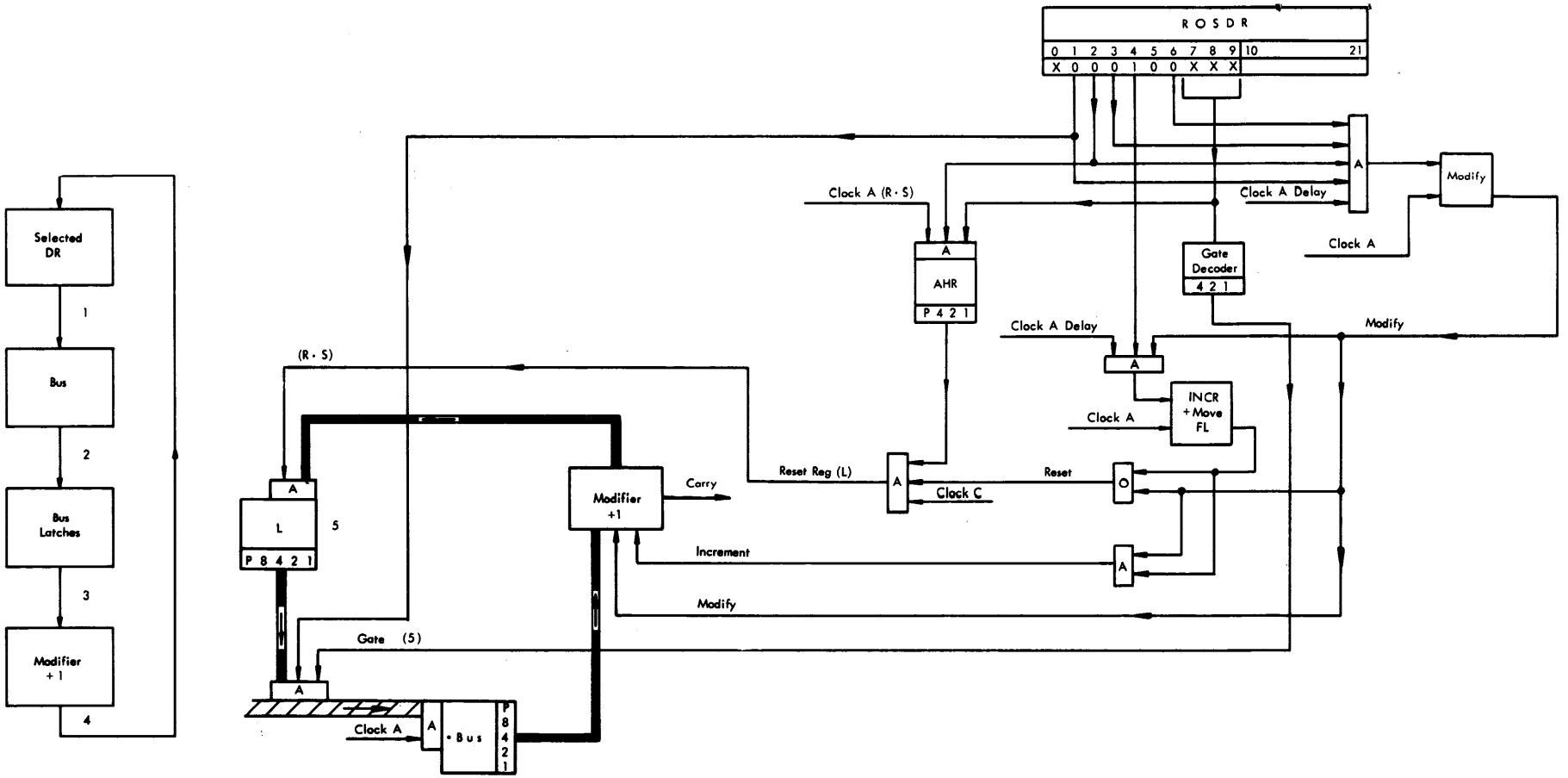
INCR X (Figure 3-6)

- Objective: Increment the selected data register (X) by 1.
- "INCR" is the micro-instruction OP code for INCRement.
- "X" is the address of the selected data register.
- The selected data register can contain binary data between 0 - 15.
- A carry occurs when the contents of the DR is 15 and an INCR micro-instruction is given. The contents of the DR is 0 after this operation.
- For a description of carry handling see section on Carry Operations.
- Figure 3-6 is a detailed diagram of the data paths and controls and includes an example to increment DR-L by 1 (9 + 1 = 10).



Register		P	ROSDR	Gate Dec		
Bit Value / Position		P 8 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1		
Content	Before	1 0 0 1 1	X 0 0 0 0 1 1 1 1 0	1 1 0	Old NSI Address 3 10 2 05	Example UXP 1010
	After	1 0 0 1 1			New NSI Address 3 10 2 06	
Remarks		3	DR-P	6=DR-P		

Figure 3-5 UXX Data Flow



Register		L	AHR	ROSDR	Gate Dec	
Bit Value / Position		P 8 4 2 1	P 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1	Example L + 1
Content	Before	1 1 0 0 1	X X X X	X 0 0 0 1 0 0 1 0 1	1 0 1	
	After	1 1 0 1 0	1 1 0 1			
Remarks		9+1=10	5=DR-L	5=DR-L	5=DR-L	

Figure 3-6 INCR X Data Flow

Circuit Objectives

ROSDR Pos. 7, 8, 9 (X X X)	are variable depending by the address of the selected DR. This address is decoded in the Gate Decoder.
ROSDR Pos. 1 (0)	is a zero (0) which is ANDed with the output of the Gate Decoder to gate the selected DR to the bus.
ROSDR Pos. 1, 2, 3, 6 0 0 0 0	are all zeros (0) and set the Modify latch.
ROSDR Pos. 4 (1)	is a one (1) which is ANDed with the Modify latch to set the Incr or Move latch. The outputs of the Modify and Incr Or Move latches are ANDed to give the Increment signal. This signal and the Modify latch cause the Modifier to increment the input from the Bus latches by 1.
ROSDR Pos. 2 (0)	is a zero (0) which sets the address of the selected DR into the AHR. The Reset signal from the Modify latch is ANDed with the output of the AHR to reset the selected DR and to set the Modifier output into the selected DR.

DECR X (Figure 3-7)

- Objective: Decrement the selected DR (X) by 1.
- "DECR" is the micro-instruction OP code for decrement.
- "X" is the address of the selected DR.
- The selected DR can contain binary data between 0 - 15.
- A carry occurs when the DR content is 0 and a DECR micro-instruction is given. The DR content is 15 after this operation.

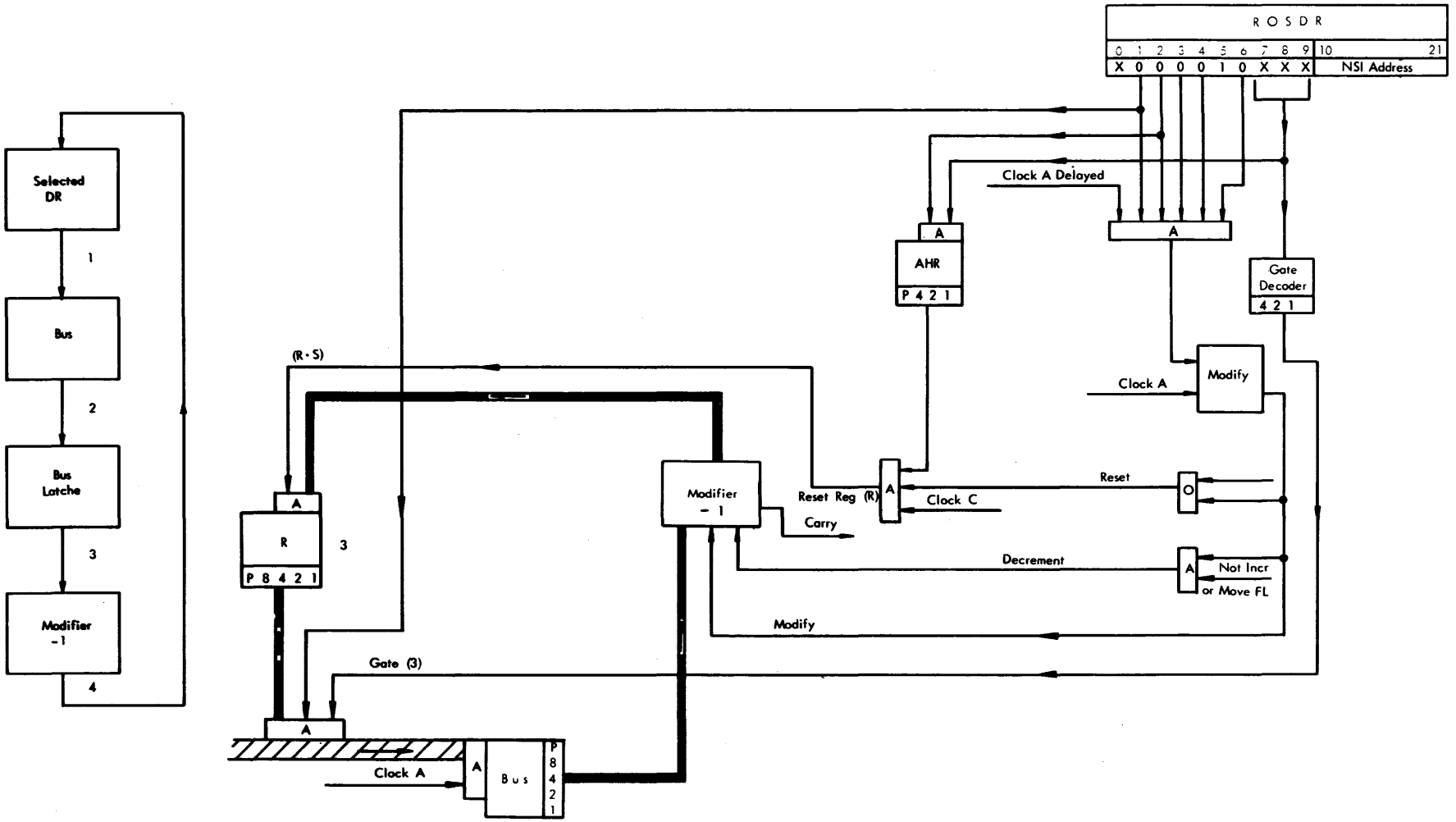
Circuit Objectives

ROSDR Pos. 7, 8, 9 (X X X)	are variable depending upon the address of the selected DR. This address is decoded in the Gate Decoder.
ROSDR Pos. 1 (0)	is a zero (0) which is ANDed with the output of the Gate Decoder to gate the selected DR to the bus.
ROSDR Pos. 1, 2, 3, 6 0 0 0 0	are all zeros (0). These zeros set the Modify latch. The output of the Modify latch ANDed with the Not output of the Incr or Move latch produces the Decrement signal. This signal and the Modify latch control the Modifier to decrement the Modifier input from the Bus latches by 1.
ROSDR Pos. 2 (0)	is a zero (0) which sets the address of the selected DR into the AHR. The Reset signal from the Modify latch is ANDed with the output of the AHR to reset and then set the selected DR with the Modifier output.

Carry Operations

- A carry occurring during an INCR X or DECR X micro-instruction operation causes the NSI to be altered thus causing a branch to occur.
- Positions 10 through 21 of the punched NSI which are in the ROS DR are, as usual, gated to positions 10 through 21 of the ROAR.
- The punched NSI in the INCR X or DECR X micro-instruction must contain a zero (0) in position 21 to enable a carry to cause a branch.
- The carry generated in the Modifier, causes a one (1) to be set in ROAR position 21 (Figure 3-8).

The new NSI address is the address of another micro-instruction which is the start of a sub-routine



Register		R	AHR	ROSDR	Gate Dec	
Bit Value / Position		P 8 4 2 1	P 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1	Example R-1
Content	Before	0 1 0 0 0	X X X X	X 0 0 0 0 1 0 0 1 1	0 1 1	
	After	0 0 1 1 1	1 0 1 1			
Remarks		8-1=7	3=DR-R	DR-R	3=DR-R	

Figure 3-7 DECR X Data Flow

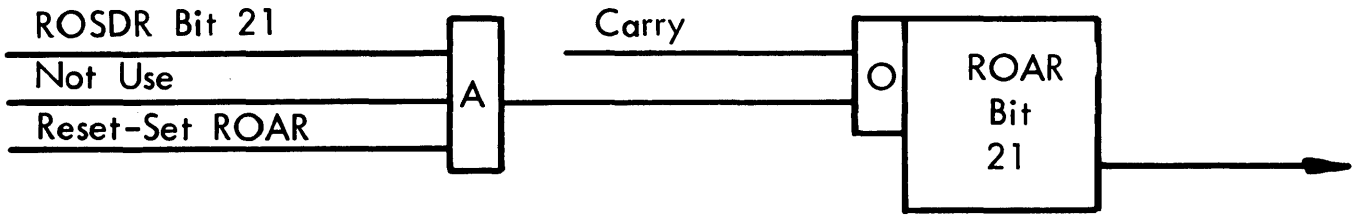


Figure 3-8 ROAR Bit 21

to update (add one or subtract one) the next sequential address, or in some way store the carry indication for later use, or possibly to use the carry to perform some action such as releasing a print hammer. The actual operation performed depends upon the particular micro-program in which an INCR X or DECR X micro-instruction is used.

F N (Figure 3-9)

- Objective: Read one Byte out of AUX storage into DR-U and L. The AUX storage is addressed by the "N" part of the micro-instruction.
- "F" is the micro-instruction Op code for Fetch.
- "N" is a binary value between 0 and 15 and is used to address a byte in the AUX storage.
- 16 different bytes in the AUX storage are accessible with the FN micro-instruction (for the location of these 16 bytes see AUX storage layout).
- The N part is set into the four lower positions of the STAR to address a byte.
- The AUX latch is set to select the AUX storage.
- Figure 3-9 is a detailed diagram of the data paths and controls and includes an example to fetch a byte (10) out of the AUX storage.

Circuit Objectives

ROSDR Pos. 6, 7, 8, 9 (X X X X) — is the N part and is variable depending upon the address of the selected Byte in the AUX storage.

ROSDR Pos. 1 (1) — is a one (1) and gates the N part of the Bus.

ROSDR Pos. 1, 2, 4 (1 0 1) — 1 and 4 are ones (1); 2 is a zero (0). This bit pattern is gated to the Op Decoder (FN). The output of the Op Decoder produces the Start Read, Storage Start, and Reset U and L Reg signals. The Reset U and L Reg signal reset the U and L DR to clear the U and L DR in preparation to receive the addressed byte of the AUX storage. The Storage Start signal is ANDed with the N part from the Bus to set positions 1, 2, 4 and 8 of the STAR.

ROSDR Pos. 3 (0) — is a zero (0) and inhibits position 16, 32, 64, and 128 of the STAR from receiving any bits. These positions are set to zeros (0) with the Storage Start signals.

ROSDR Pos. 1 (1) — is a one (1) and inhibits positions 256, 512, 1024, 2048, and 4096 of the STAR from receiving any bits. These positions are set to zeros (0) with the Storage Start signal.

The Start Read signal gates the STAR output through the STAR Decoder to activate the AUX storage and to read out the addressed byte into DR-U and L.

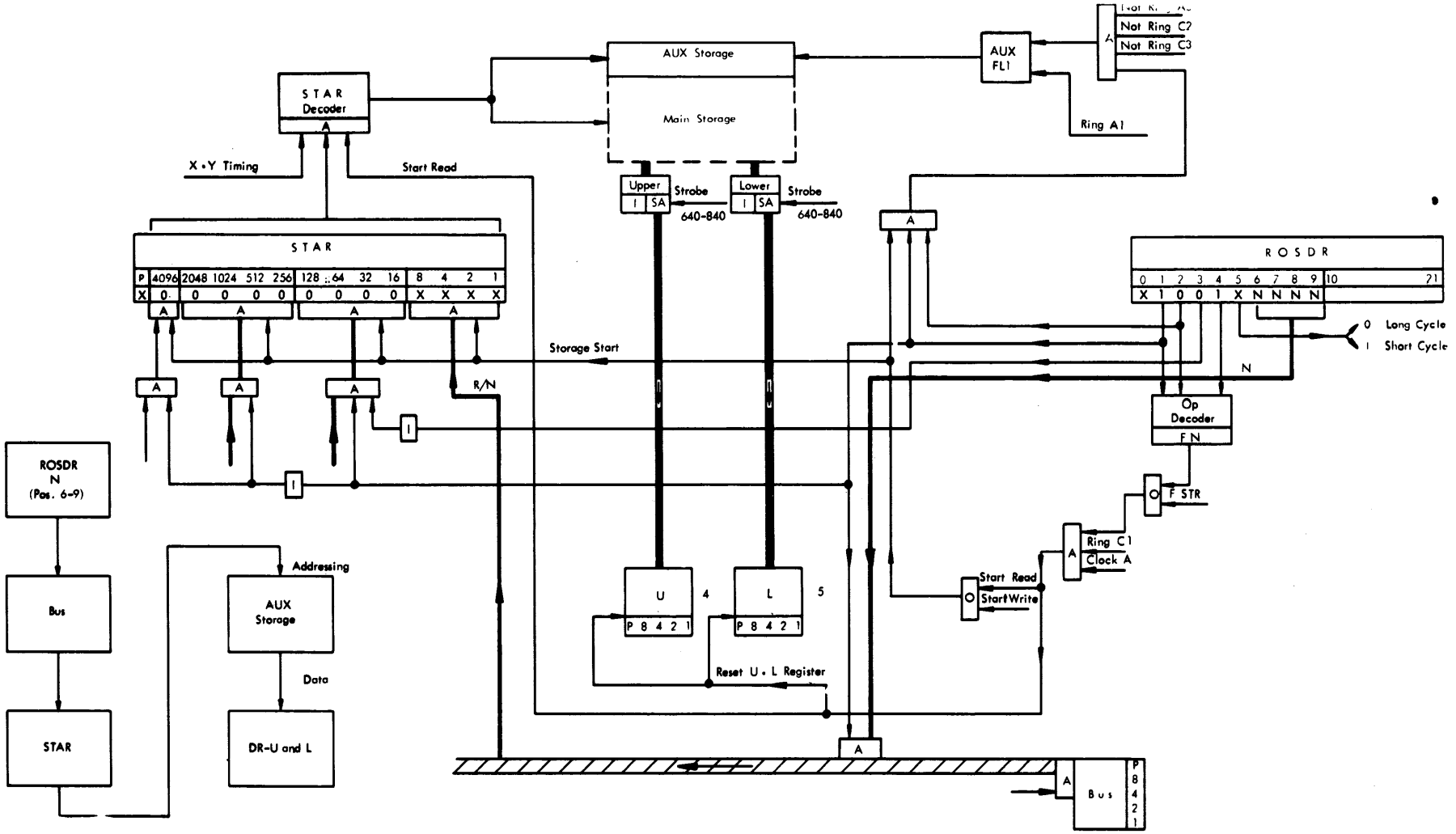
ROSDR Pos. 1, 2 (1 0) — 1 is a one (1); 2 is a zero (0). This bit pattern ANDed with the Storage Start signal sets the AUX latch. The AUX latch selects the AUX storage.

ROSDR Pos. 5 (X) — can be either a zero or a one, depending on whether a "long cycle" or a "short cycle" is desired. Long and short cycle timings and objectives are described in Section ROAR.

All lines labeled "RING..." are dependent upon long or short cycle timing.

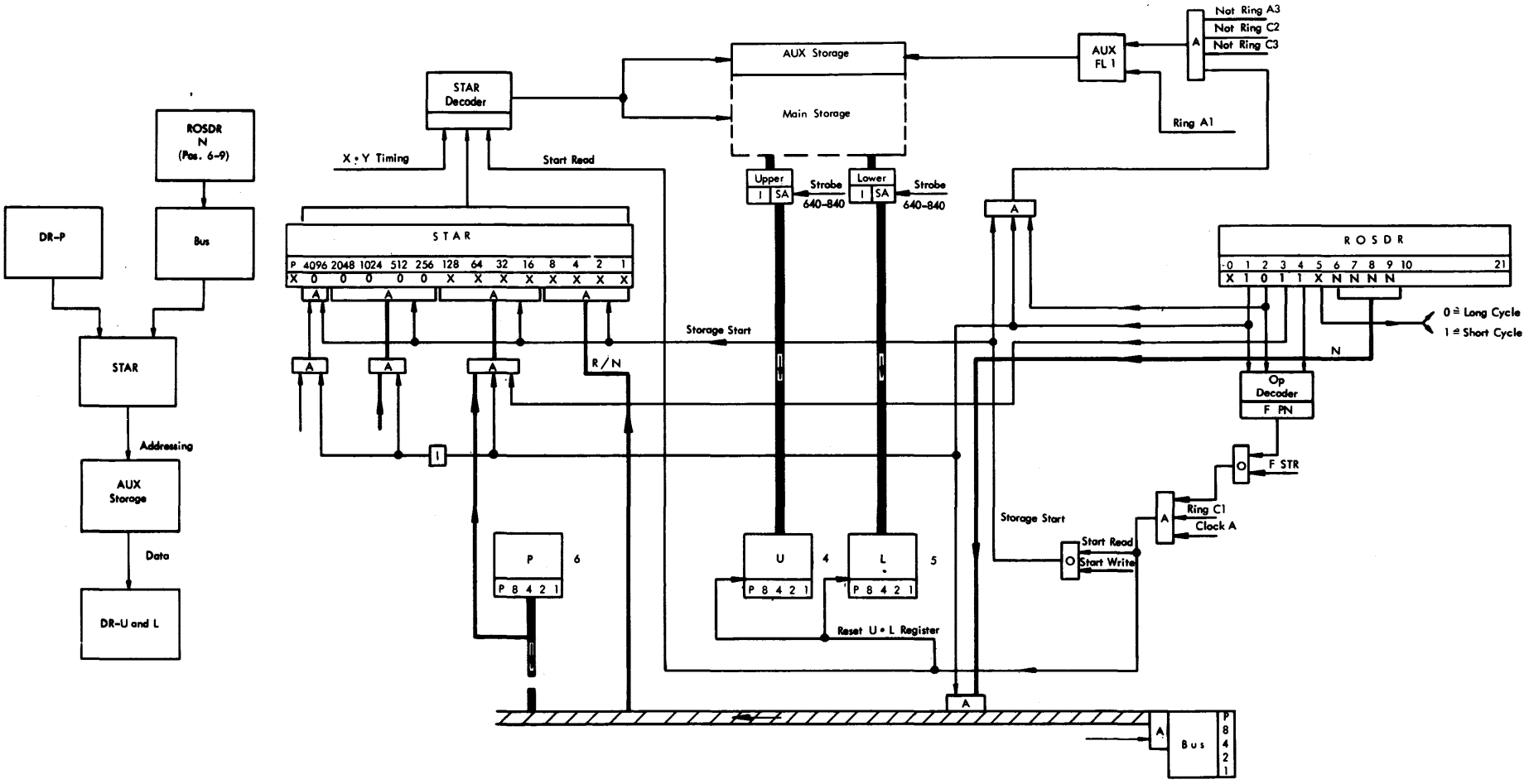
F PN (Figure 3-10)

- Objective: Read one byte out of AUX storage into DR-U and L. The AUX storage is addressed by the contents of DR-P (previously loaded by an MX*N micro instruction) and the N part of the FPN micro-instruction.
- "F" is the micro-instruction Op code for Fetch.



Register		STAR													U	L	ROSDR		
Bit Value / Position		P	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	P 8 4 2 1	P 8 4 2 1	0 1 2 3 4 5 6 7 8 9	
Content	Before	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X X X X X	X X X X X	X 1 0 0 1 X 1 0 1 0	Example F 10
	After	1	0	0	0	0	0	0	0	0	0	1	0	1	0	Y Y Y Y Y	Y Y Y Y Y		
Remarks		Byte /0A/																N=10	

Figure 3-9 FN Data Flow



Register		STAR													P	U	L	ROSDR	
Bit Value / Position		P	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	0 1 2 3 4 5 6 7 8 9
Content	Before	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0 1 0 1 1	X X X X X	X X X X X	X 1 0 1 1 X 0 1 0 1
	After	0	0	0	0	0	0	1	0	1	1	0	1	0	1		Y Y Y Y Y	Y Y Y Y Y	
Remarks		Byte /B5/														/B/			N=5

Example F P5

Figure 3-10 F PN Data Flow

- "P" designates DR-P the content of which is used to address in part the AUX storage.
- N is a four bit part of the micro-instruction which is used, together with the contents of DR-P, to address a byte in the AUX storage.
- The N part and the contents of DR-P are set into the STAR to address a byte.
- The AUX latch is set to select the AUX storage.
- All 256 bytes are accessible in the AUX storage with the FPN micro-instruction.
- Figure 3-10 is a detailed diagram of the data paths and controls and includes an example to fetch a byte (/B5/) out of the AUX storage.

Circuit Objectives

ROSDR Pos. 6, 7, 8, 9 (X X X X) — is the N part and is variable depending upon the address of the selected byte in the AUX storage.

ROSDR Pos. 1 (1) — is a one (1) and gates the N part to the Bus.

ROSDR Pos. 1, 2, 4 (1 0 1) — 1 and 4 are ones (1); 2 is a zero (0). This bit pattern is gated to the Op Decoder (FPN). The output of the Op-Decoder provides the Start Read, Storage Start, and the Reset U and L Reg signals.

The Reset U and L Reg signal resets the U and L DR to ensure that the U and L DR is empty and ready to receive the addressed byte of the AUX storage.

The Storage Start signal is ANDed with the N part from the Bus to set positions 1, 2, 4, and 8 of STAR.

ROSDR Pos. 1, 3 (1 1) — are ones (1) and are ANDed with the output of the DR-P. The output of these AND switches are ANDed with the Storage Start signal to set positions 16, 32, 64, and 128 of STAR.

ROSDR Pos. 1 (1) — is a one (1) and inhibits positions 256, 512, 1024, 2048, and 4096 of STAR from receiving any bits. These positions are set to zeros (0) with the Storage Start signal.

The Start Read signal gates the STAR output through the STAR Decoder to activate the AUX storage and to read out the addressed byte into DR-U and L.

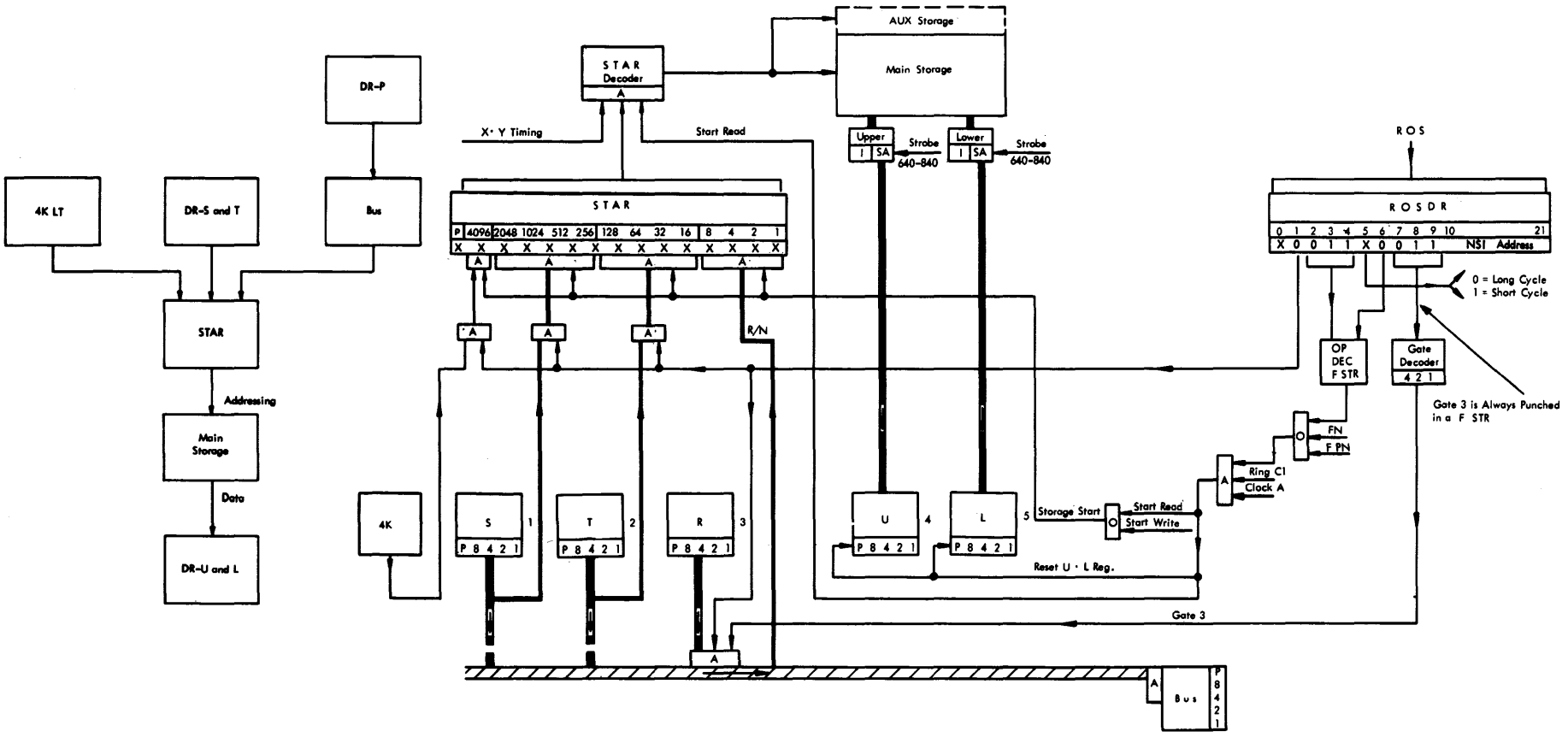
ROSDR Pos. 1, 2 (1 0) — 1 is a one (1); 2 is a zero (0). This bit pattern ANDed with the Storage Start signal sets the AUX latch. The AUX latch selects the AUX storage.

ROSDR Pos. 5 (X) (0/1) — can be either a zero or a one, depending on whether a "long cycle" or a "short cycle" is desired. Long and short cycle timings and objectives are described in Section ... ROAR.

All lines labeled "Ring..." are dependent upon long or short cycle timing.

F STR (Figure 3-11)

- Objective: Read one byte out of Main storage into the U and L data registers.
- "F" is the micro-instruction OP code for Fetch.
- "STR" designates the DR-S, T, and R the contents of which are used to address the Main storage.
- DR-S, T, and R are set into STAR to address a byte in the Main storage.
- DR-S, T, and R contains a total of 12 bits, which permit a maximum of 4096 addressable bytes in Main storage (4K storage).
- Addressing an 8K storage requires a 13th bit.
- The "K" latch contains the 13th bit which is set into STAR during the F STR micro-instruction.
- To place an address into the latch and into the DR S, T, and R, a special micro-program subroutine is used.



Register		STAR	4K	S	T	R	U	L	ROSDR	Gate Dec	
Bit Value / Position		P 4096 2048 1024 512 256 128 64 32 16 8 4 2 1		P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1	
Content	Before	X X X X X X X X X X X X X X X	1	0 0 0 0 1	0 1 1 0 1	0 0 1 1 1	X X X X X	X X X X X	X 0 0 1 1 X 0 0 1 1	0 1 1	Example F STR
	After	1 1 0 0 0 1 1 1 0 1 0 1 1 1		0 0 0 0 1	0 1 1 0 1	0 0 1 1 1	Y Y Y Y Y	Y Y Y Y Y			
Remarks		Byte /11D7/	/1/	/1/	/D/	/7/			DR-R	3 = DR-R	

Figure 3-11 F STR Data Flow

- Figure 3-11 is a detailed diagram of the data paths and controls and includes an example to fetch a byte (/11D7/) out of the Main storage.

Circuit Objectives

ROSDR Pos. 7, 8, 9 (0 1 1) — 7 is a zero (0); 8 and 9 are ones (1). This bit pattern is decoded in the Gate Decoder as Gate 3 which is used with ROSDR Pos. 1.

ROSDR Pos. 1 (0) — is a zero (0) which is ANDed with Gate 3 to gate DR-R to the bus.

This 0 is also ANDed with the output of the latch and DR-S and T to gate these registers to STAR.

ROSDR Pos. 2, 3, 4, 6 (0 1 1 0) — 2 and 6 are zeros (0); 3 and 4 are ones (1). These bits are decoded in the Op Decoder as a F STR micro-instruction. The output of the Op Decoder with timings, provide these signals: "Start Read", "Storage Start", and Reset U and L Reg".

The Reset U and L Reg signal resets the U and L DR to ensure that the U and L DRs are empty, to receive the addressed bytes of the main storage. The Storage Start signal sets the contents of the K, S, T, and R registers to STAR.

The Storage Start Read signal gates the output of the STAR through the STAR Decoder to activate main storage and to read out the addressed byte.

ROSDR Pos. 5 (X) — can be either a zero or a one, depending on whether a "long cycle" or a "short cycle" is desired. Long and short cycle timings and objectives are described in Section ... ROAR.

All lines labeled "Ring..." are dependent upon long or short cycle timing.

S N (Figure 3-12)

- Objective: Write one byte into AUX storage from U and L DRs.
- The AUX storage is addressed by the "N" part of the micro-instruction.
- "S" is the micro-instruction Op code for Store.
- "N" is a binary value between 0 and 15 and is used to address a byte in the AUX storage.
- 16 different bytes in the AUX storage are accessible with the SN micro-instruction for the location of the 16 bytes.
- The N part is set into the four lower positions of the STAR to address a byte.
- The AUX latch is set to select the AUX storage.

- Figure 3-12 is a detailed diagram of the data paths and controls, and includes an example to store a Byte (14) into the AUX storage.

Circuit Objectives

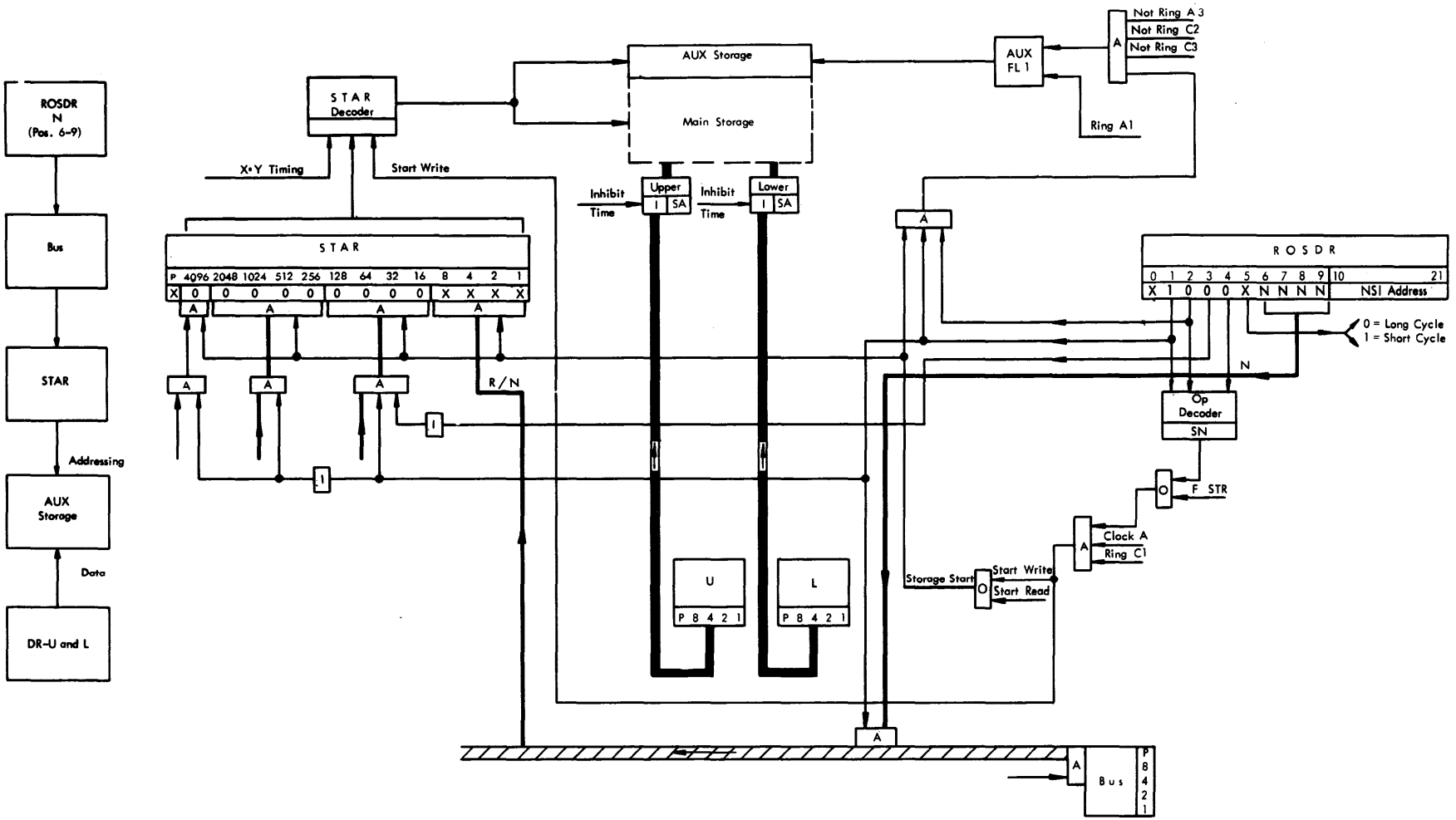
ROSDR Pos. 6, 7, 8, 9 (X X X X) — is the N part and is variable depending upon the address of the selected Byte in the AUX storage.

ROSDR Pos. 1 (1) — is a one (1) and gates the N part to the Bus.

ROSDR Pos. 1, 2, 4 (1 0 1) — 1 and 4 are ones (1); 2 is a zero (0). This bit pattern is gated to the Op Decoder (SN). The output of the Op Decoder produces the Start Write and the Storage Start signals.

The Storage Start signal is ANDed with the N part from the Bus to set positions 1, 2, 4, and 8 of the STAR.

ROSDR Pos. 3 (0) — is a zero (0) and inhibits positions 16, 32, 64, and 128



Register		STAR														U	L	ROSDR		
Bit Value / Position		P	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	P 8 4 2 1	P 8 4 2 1	0 1 2 3 4 5 6 7 8 9		
Content	Before	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X X X X X	X X X X X	X 1 0 0 1 X 1 1 1 0		Example S 14
	After	0	0	0	0	0	0	0	0	0	0	1	1	1	0					
Remarks		Byte /0E/																N=14		

Figure 3-12 SN Data Flow

of the STAR from receiving any bits. These positions are set to zeros (0) with the Storage Start signal.

ROSDR Pos. 1 (1) is a one (1) and inhibits positions 256, 512, 1024, 2048 and 4096 of the STAR from receiving any bits. These positions are set to zeros (0) with the Storage Start signal.

The Start Write signal gates the STAR output through the STAR Decoder to activate the AUX storage and to write the addressed byte into the AUX storage.

ROSDR Pos. 1, 2 (1 0) 1 is a one (1); 2 is a zero (0). This bit pattern, ANDed with the Storage Start signal, sets the AUX latch. The AUX latch selects the AUX storage.

ROSDR Pos. 5 (X) can be either a zero or a one, (1), depending on whether a "long cycle" or a "short cycle" is desired. Long and short cycle timings and objectives are described in Section ... ROAR.

All lines labeled "Ring..." are dependent upon long or short cycle timing.

S PN (Figure 3-13)

- Objectives: Write one byte into AUX storage from U and L data registers.
- The AUX storage is addressed by the contents of DR-P and the N part of the micro-instruction.
- "S" is the micro-instruction Op code for Store.
- "P" designates DR-P, the contents of which are used to address in part the AUX storage.
- "N" is a four bit part of the micro-instruction, which is used, together with the contents of DR-P, to address a byte in AUX storage.
- The N part and the contents of DR-P are set into STAR to address a byte.
- The AUX latch is set to select the AUX storage.
- All 256 bytes in AUX storage are accessible with the FPN micro-instruction.
- DR-P is previously loaded with a MX*N or MX*X micro-instruction.
- Figure 3-13 is a detailed diagram of the data paths and controls, and includes an example to store a byte (14) into AUX storage.

Circuit Objectives

ROSDR Pos. 6, 7, 8, 9 (X X X X) is the N part and is variable depending upon the address of the selected Byte in the AUX storage.

ROSDR Pos. 1 (1) is a one (1) and gates the N part to the Bus.

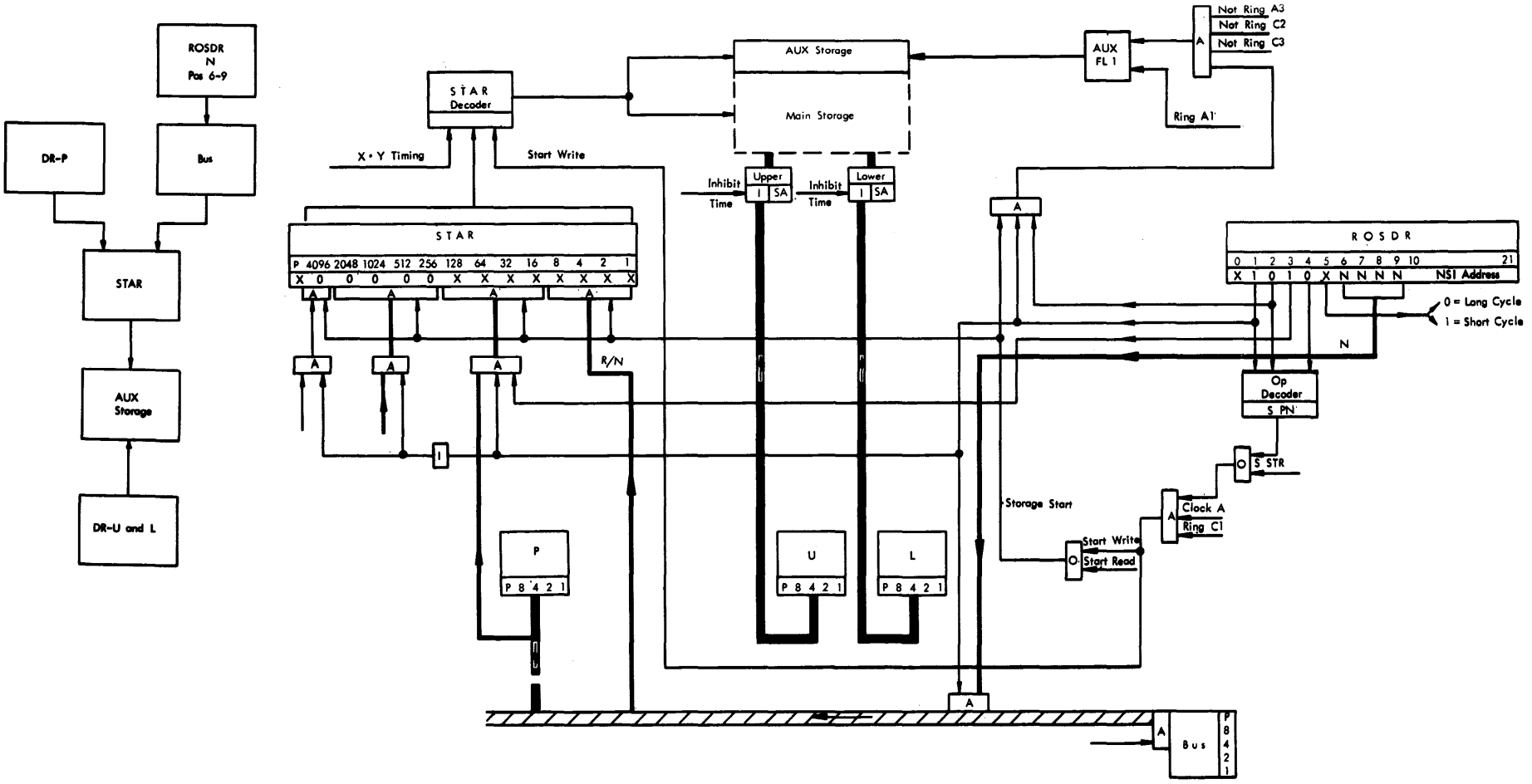
ROSDR Pos. 1, 2, 4 (1 0 1) 1 and 4 are ones (1); 2 is a zero (0). This bit pattern is gated to the Op Decoder (FPN). The output of the Op Decoder provides the Start Write and Storage Start signals.

The Storage Start signal is ANDed with the N part from the Bus to set positions 1, 2, 4 and 8 of STAR.

ROSDR Pos. 1, 3 (1 1) are ones (1). These bits are ANDed with the output of DR-P and the outputs of these AND AND switches are ANDed with the Storage Start signal to set positions 16, 32, 64, and 128 of the STAR.

ROSDR Pos. 1 (1) is a one (1) and inhibits positions 256, 512, 1024, 2048, and 4096 of STAR from receiving any bits. These positions are set to zeros (0) with the Storage Start signal.

The Start Write signal gates the STAR output through the STAR decoder to activate the AUX storage and to write the addressed byte into the AUX storage.



Register		STAR														P	U	L	ROSDR										Example S P7
Bit Value / Position		P	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	0	1	2	3	4	5	6	7	8	9	
Content	Before	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1 0 1 0 1	X X X X X	X X X X X	X	1	0	1	0	X	0	1	J	1		
	After	0	0	0	0	0	0	0	1	0	1	0	1	1	1														
Remarks		Byte /57/														/5/												N = 7	

Figure 3-13 S PN Data Flow

ROSDR Pos. 1, 2 (1 0) — 1 is a one (1), 2 is a zero (0). This bit pattern ANDed with the Storage Start signal sets the AUX latch which selects the AUX storage.

ROSDR Pos. 5 — can be either a zero or a one, depending on whether a "long cycle" or a "short cycle" is desired. Long and short cycles timing and objectives are described in Section ... ROAR.

All lines labeled "Ring..." are dependent upon long or short cycle timing.

S STR (Figure 3-14)

- Objective: Write one Byte into the Main storage from U and L data registers.
- "S" is the micro-instruction Op code for Store.
- "STR" designates the DR-S, T, and R, the contents of which are used to address the Main storage.
- DR-S, T, and R contain a total of 12 bits, which permit a maximum of 4096 addressable bytes in Main storage (4K storage).
- Addressing an 8K storage requires a 13th bit.
- The DR-A contains the 13th bit which is set into STAR during the F STR micro-instruction.
- A special micro-program sub-routine was used previously to place an address into the DR-S, T, and R.

Circuit Objectives

ROSDR Pos. 7, 8, 9 (0 1 1) — 7 is a zero (0); 8 and 9 are ones (1). This bit pattern is decoded in the Gate Decoder as Gate 3 which is used with ROSDR POS 1.

ROSDR Pos. 1 (0) — is a zero (0) which is ANDed with Gate 3 to Gate DR-R to the Bus.

This 0 is also ANDed with the output of DR-S and T to gate these registers to STAR.

ROSDR Pos. 2, 3, 4, 6 (0 1 0 0) — 2, 4, and 6 are zeros (0), 3 is a one (1). These bits are decoded in the Op Decoder as a F STR micro-instruction. The output of the Op Decoder with timings, provide the following signals: "Start Write" and "Storage Start."

The Storage Start signal sets the content of the 4K latch, S, T, and R registers into STAR.

The Start Write signal gates the output of STAR through the STAR Decoder to activate Main storage and to write the addressed bytes into Main storage.

ROSDR Pos. 5 (X) — can be either a zero or a one, depending on whether a "long cycle" or a "short cycle" is desired. Long and short cycle timings and objectives are described in Section ... ROAR.

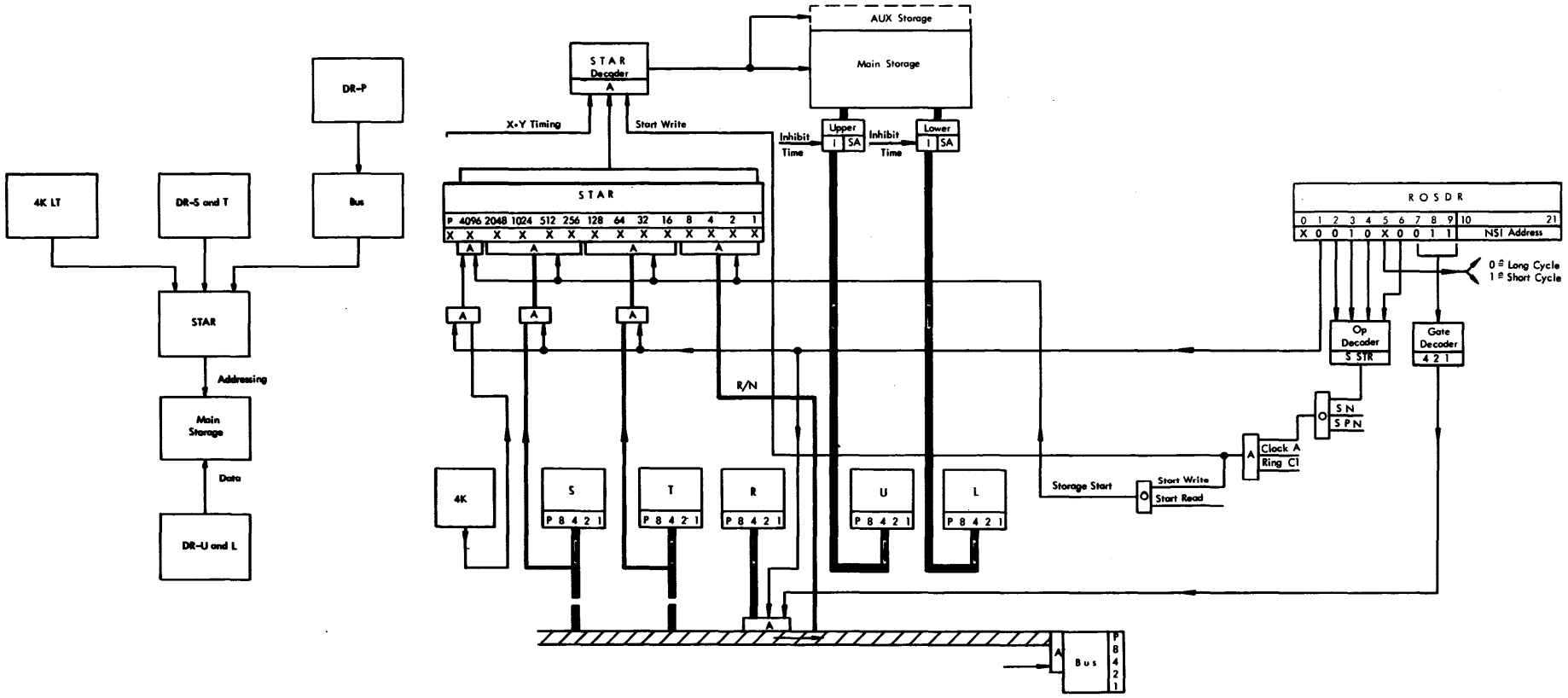
All lines labeled "Ring..." are dependent upon long or short cycle timing.

Sense (Figure 3-15)

- Objectives: Sense the condition of various CPU or I/O Indicators and set the sensed condition into DR-L.
- 64 sense addresses are available and are selected with the 6 position "n" field in the micro-instruction. See Sense Instructions.
- Each Sense address is able to set up four Indicators into DR-L.
- In DR-L, one indicator is set as bit 1, the other indicators are set as bit 2, 4, and 8 (Figure M31).

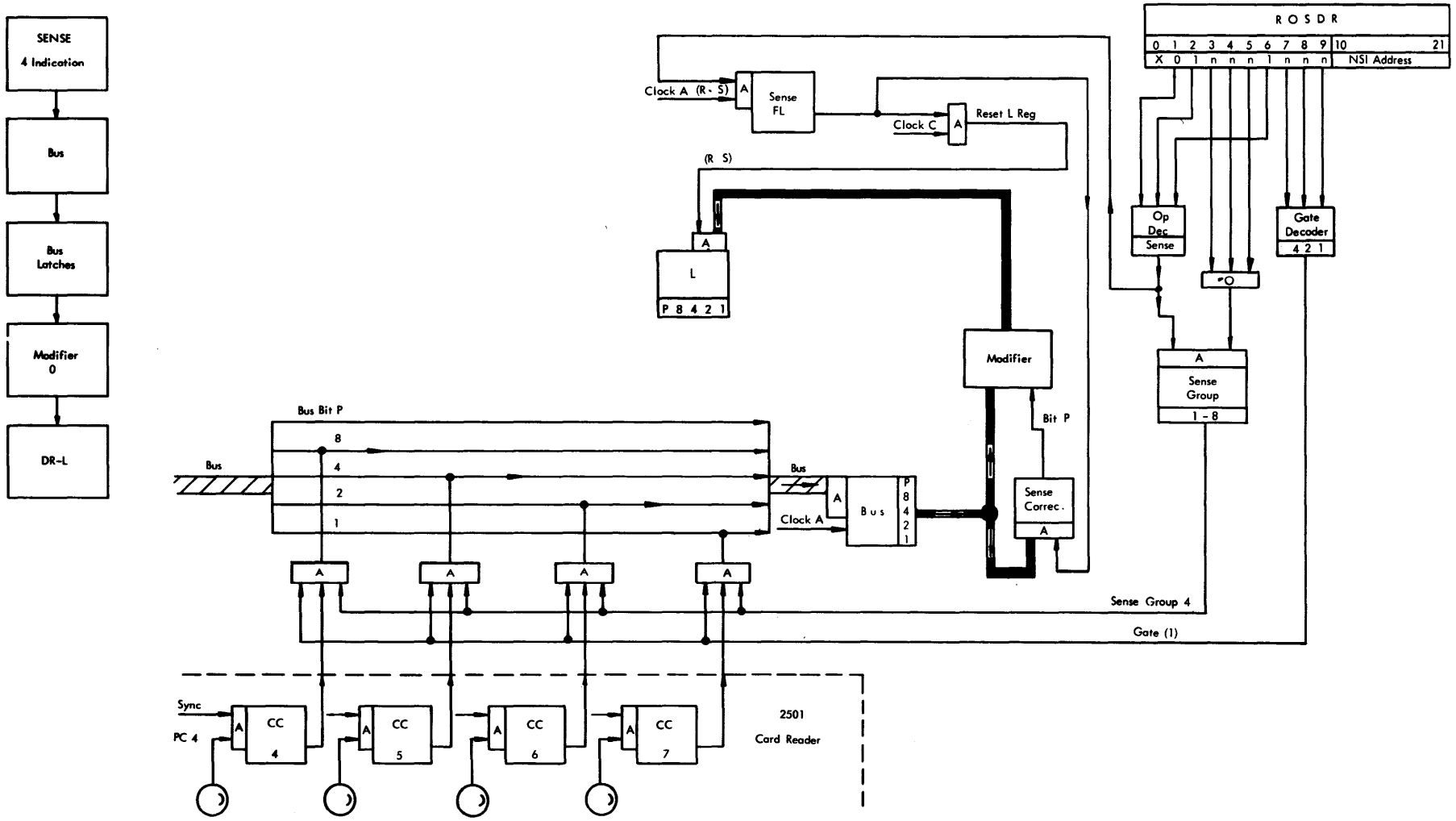
Description

When a micro-program is in progress, the SENSE micro-instruction provides the only means of testing for circuit conditions. Circuit conditions are items such as data read from a card, the stop latch in the CPU being ON, or the MODE switch being in the ADDRESS STOP position. There are 64 different



Register		STAR	4K	S	T	R	U	L	ROSDR	Gate Dec	
Bit Value / Position		P 4096 2048 1024 512 256 128 64 32 16 8 4 2 1		P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	0 1 2 3 4 5 6 7 8 9	4 2 1	Example S STR
Content	Before	X X X X X X X X X X X X X X	0	1 0 0 1 1	0 1 1 1 0	0 1 0 0 0	X X X X X	X X X X X	X 0 0 1 0 X 0 0 1 1	0 1 1	
	After	1 0 0 0 1 1 1 1 1 0 1 0 0 0									
Remarks		Byte /03E8/	/0/	/3/	/E/	/8/				DR-R	3 = DR-R

Figure 3-14 S STR Data Flow



Register		CC 4	CC 5	CC 6	CC 7	L	Bus FL	ROSDR	Sense Group	Gate Dec	
Bit Value / Position						P 8 4 2 1	P 8 4 2 1	0 1 2 3 4 5 6 7 8 9	32 16 8	4 2 1	
Content	Before	On	Off	On	Off	X X X X X	X X X X X	X 0 1 0 1 1 1 1 0 0 1	0 1 1	0 0 1	Example Sense 25
	After					1 1 0 1 0	0 1 0 1 0				
Remarks		2501 Card Reader						Group 3 Gate 1	3	1	

Figure 3-15 Sense Data Flow

SENSE addresses capable of simultaneously setting up to four bits (conditions) on the Bus. One condition is set as bit 1, the next with bit 2, then bit 4 finally bit 8. The bus bits are gated, via the Bus latch, through the Modifier into the DR-L.

Later in the micro-program, the bits in DR-L are tested with a USE micro-instruction to select and test the one particular indicator desired to determine if a branch in the micro-program is required. Figure 3-16 illustrates an address block (SENSE 0) and shows how the 4 Indicators are related to the 4 bus lines; the right hand indicator (in the block) is related to bus bit one, the next left to bus bit two etc. A bus line can be set if the Indicator is On or Off. Figure 3-16 shows that the 8 bit in a SENSE 0 is set if the "Mode" switch is NOT in Storage SCAN position and a 4 bit is set when the Mode switch is in the DISPLAY position.

All Indicators which can be sensed are shown in the Sense Instructions Table.

Circuit Objectives

ROSDR Pos. 1, 2, 6 (0 1 1) — 1 is a zero (0); 2 and 6 are ones (1). This bit pattern is decoded in the Op Decoder (SENSE). The output of the Op Decoder sets the "Sense" latch.

ROSDR Pos. 3, 4, 5, 7, 8, 9 (n n n n n n) 32 16 8 4 2 1 — indicated by "n", are depending upon the 64 possible Sense addresses.

Example: Sense address 43.

ROSDR POS	3	4	5	7	8	9
Values of n	32	16	8	4	2	1
Bits		1	0	1	0	1

ROSDR Pos. 3, 4, 5 (n n n) — are ANDed with the output of the Op Decoder and decoded into "Sense Group 1-8".

ROSDR Pos. 7, 8, 9 (n n n) — are decoded in the Gate Decoder to "gate 0-7".

A sense Group and a gate are ANDed with up to 4 Indicators, to activate bus lines 1, 2, 4, and 8 when the Indicators are ON.

If it is necessary, a P (Parity) bit is generated in the "Sense Bit Correction" circuit by the output of the Bus latches ANDed with the Sense latch.

The parity bit and the bits in the Bus latches (1, 2, 4, or 8) are gated through the modifier (without changing) to the DR-L.

The output of the Sense latch is ANDed with Clock C, to reset DR-L and set the output of the Modifier into DR-L.

CTL (Control) (Figure 3-17)

- Objectives: Initiate a CPU or I/O operation such as stop the CPU or release the Card Reader clutch.
- 32 control addresses are available and are selected with the five position "n" field. See Control Instructions Table
- There are two types of control instructions:
  1. An unconditional Control instruction which executes a function independent of any other condition or register content.
  2. A conditional Control instruction which executes a function depending upon the contents of the U or L data register.

Description

When a micro-program is in progress, the CTL micro instruction provides the only means of changing a circuit condition. A CTL 0 for example, can stop the CPU by resetting the Process latch.

An unconditional Control instruction initiates the control operation without any other condition being required. For example, CTR 0 resets the "Process latch" in the CPU.

A conditional Control instruction initiates a control operation only when a certain bit is in the DR-U or L. For example, to release the clutch of the IBM 2501 Card Reader and to read a card, a control 12 must be given in the micro program and DR-L must contain a 4 bit.

Circuit Objectives

ROSDR Pos. 4, 5, 7, 8, 9 (n n n n n) — indicated by "n" are variable depending upon the 64 possible CTL addresses.

ROSDR Pos. 1, 2, 3, 4, 6 (0 0 1 u 1) — 1 and 2 are zeros (0); 3 and 6 are ones (1), 4 is variable depending upon the

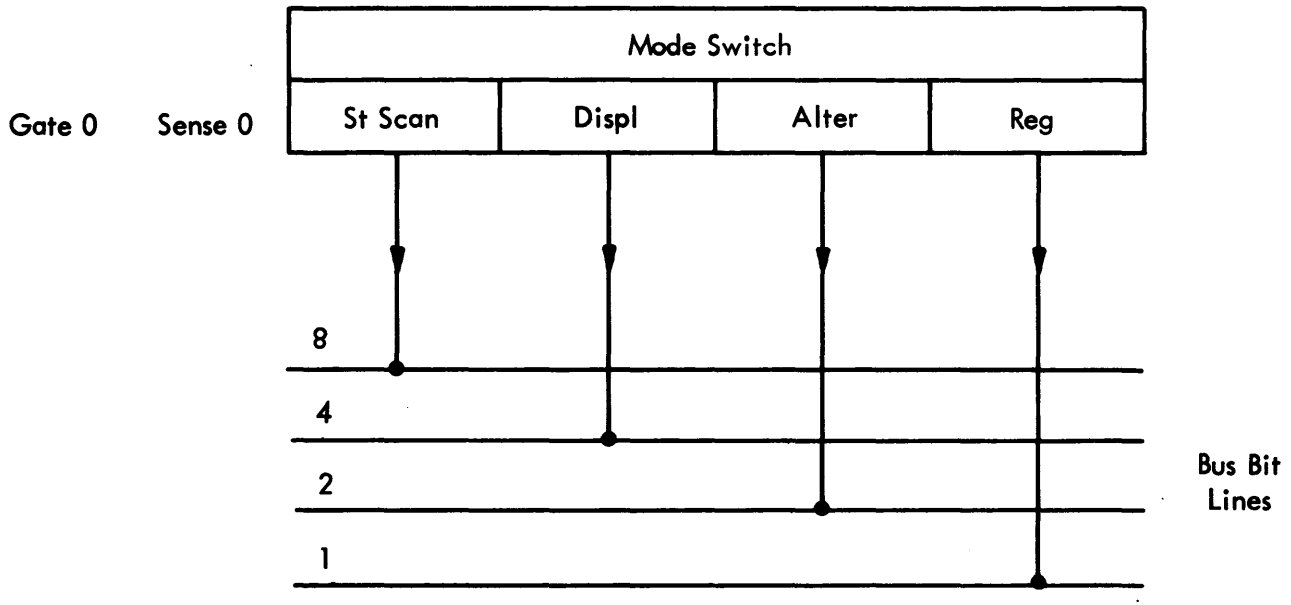
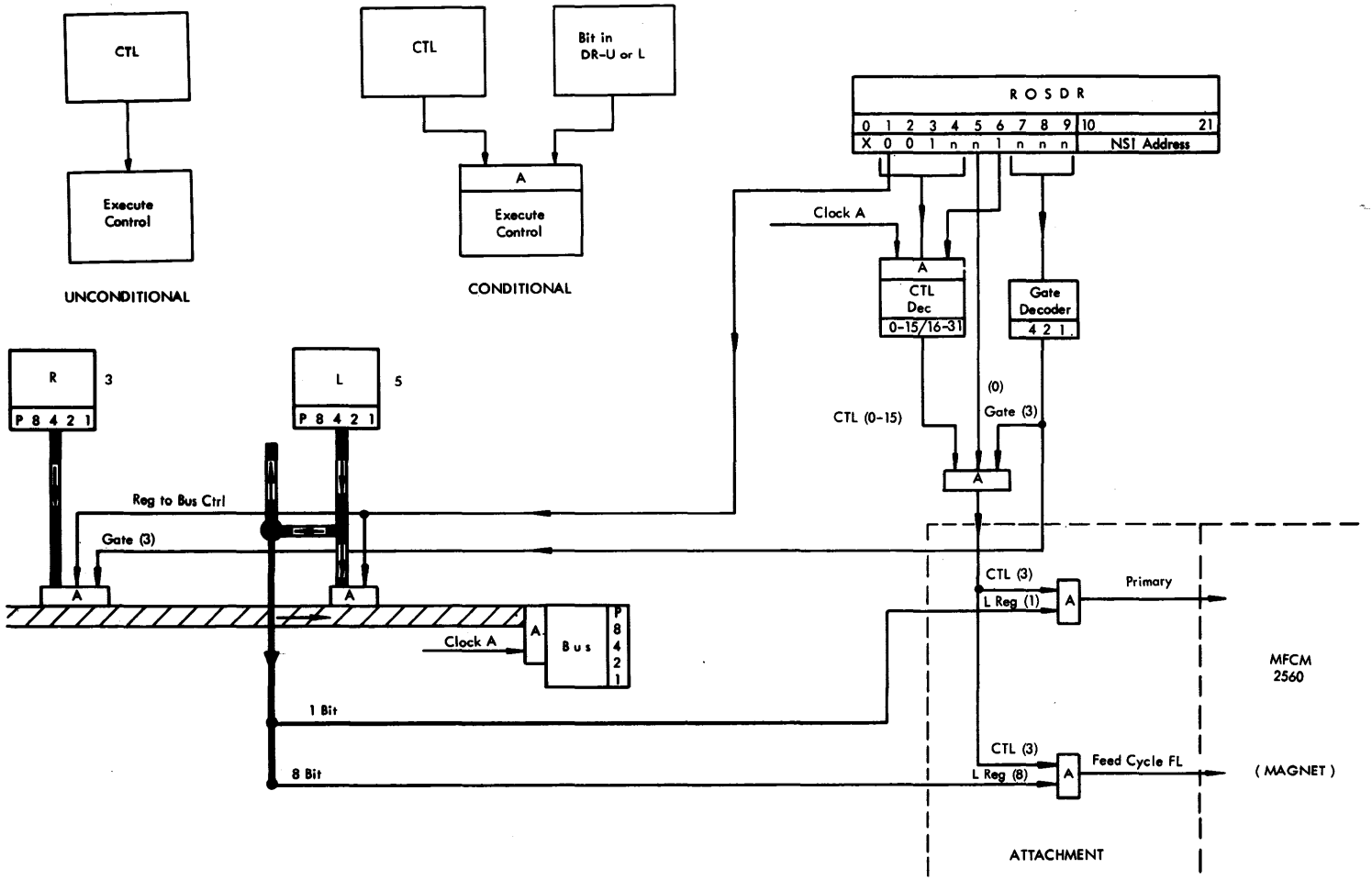


Figure 3-16 Relation of Sense Indicators to Bus Lines



Register		R	L	Bus FL	CTL Dec	ROSDR	Gate Dec	
Bit Value / Position		P 8 4 2 1	P 8 4 2 1	P 8 4 2 1	0-15/16-31	0 1 2 3 4 5 6 7 8 9	4 2 1	EXAMPLE : CONTROL 3 MFCM (2560) PRIMARY FEED READ
Content	Before	X X X X X	1 1 0 0 1	Y Y Y Y Y	0-15	X 0 0 1 0 0 1 0 1 1	0 1 1	
	After	X X X X X	1 1 0 0 1	X X X X X				
Remarks						CTL Gate 3	Gate 3	

Figure 3-17 Control Data Flow

CTL micro-instruction. This bit pattern with position 4 a zero (0) is decoded in the Control Decoder as CTL 0-15. The bit pattern with position 4 a one (1) is decoded in the Control Decoder as CTL 16 - 31.

ROSDR Pos. 7, 8, 9 (n n n) — are variable depending upon the Control instruction. These bits are decoded in the Gate Decoder.

ROSDR Pos. 5 (n) — is variable depending upon the selected Control instruction. This position ANDed with the output of the Control Decoder and Gate Decoder generates the selected CTL micro-instruction signal.

ROSDR Pos. (1) (0) — is a zero (0) and is ANDed with the output of the Gate Decoder and gates a DR (depending upon the Gate Decoder output) to the Bus.

These have no effect on the CTL instruction, but the contents of the Bus latches is changed after a CTL micro-instruction.

MICRO-PROGRAM SUBROUTINES

- Subroutines are used to perform a certain job within a main micro-program.
- A main micro-program is a E-phase of any machine Op code.
- A Subroutine is entered from various main micro-programs and branches back into the same main micro-program from which the subroutine is entered (Figures 3-18 and 3-19).

Description

As an example, the Adder subroutine is used for all arithmetic operations. During an Add Fixed Point (RR or RX format micro-program) the Adder subroutine is used to add the contents of two DRs together. A /0/ is moved into DR-I by a MX N micro-instruction to prepare the NSI address to be used as a branch back address when the subroutine is completed and the main micro-program is again resumed.

NOTE: The last micro-instruction of all main micro-programs being performed before the subroutine is entered have the same NSI address except in the Mask position. (Address of the first micro-instruction in the Subroutine.)

The last micro-instruction in the Adder subroutine is a UOI. The Mask of this Use micro-instruction is 0000 and depending upon the contents of DR-I the NSI address is modified to return to the proper main micro-program (Figure 3-19).

All subroutines (except one) are entered and left in the same manner as described for the Adder subroutine.

Adder

- Objective: Perform all arithmetic operations by adding the binary content of DR-R to the binary content of DR-L.
- Before the Adder micro program subroutine can be used, the Operand 1 values must be moved into DR-L and Operand 2 values must be moved into DR-R.
- Binary: If more than four positions are to be added the Adder subroutine is repeated.
- A carry from one addition to the next is stored in DR-E 1 bit position and is processed the next time the adder subroutine is used.
- The result is always in DR-L and the content of DR-R is destroyed.
- Decimal: If more than one decimal position is to be added the adder subroutine is repeated.

Description

Adding two values (represented in binary) by means of the Adder subroutine is accomplished as follows:

One value (operand) is decremented by 1, the other value (operand) is incremented by 1. The incrementing and decrementing is performed until a carry occurs (from 0 to 15) in the Operand which is decremented.

When a carry occurs from the decremented Operand, the value in the incremented Operand is the sum of the two values.

Incrementing by 1 and decrementing by 1 is accomplished with INCR X and DECR X micro-instructions.

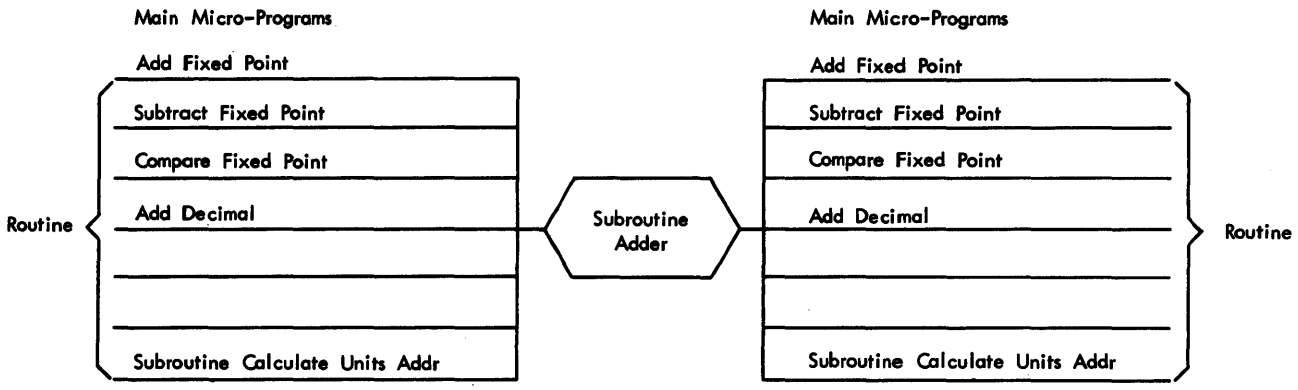


Figure 3-18 Principle of Using a Subroutine

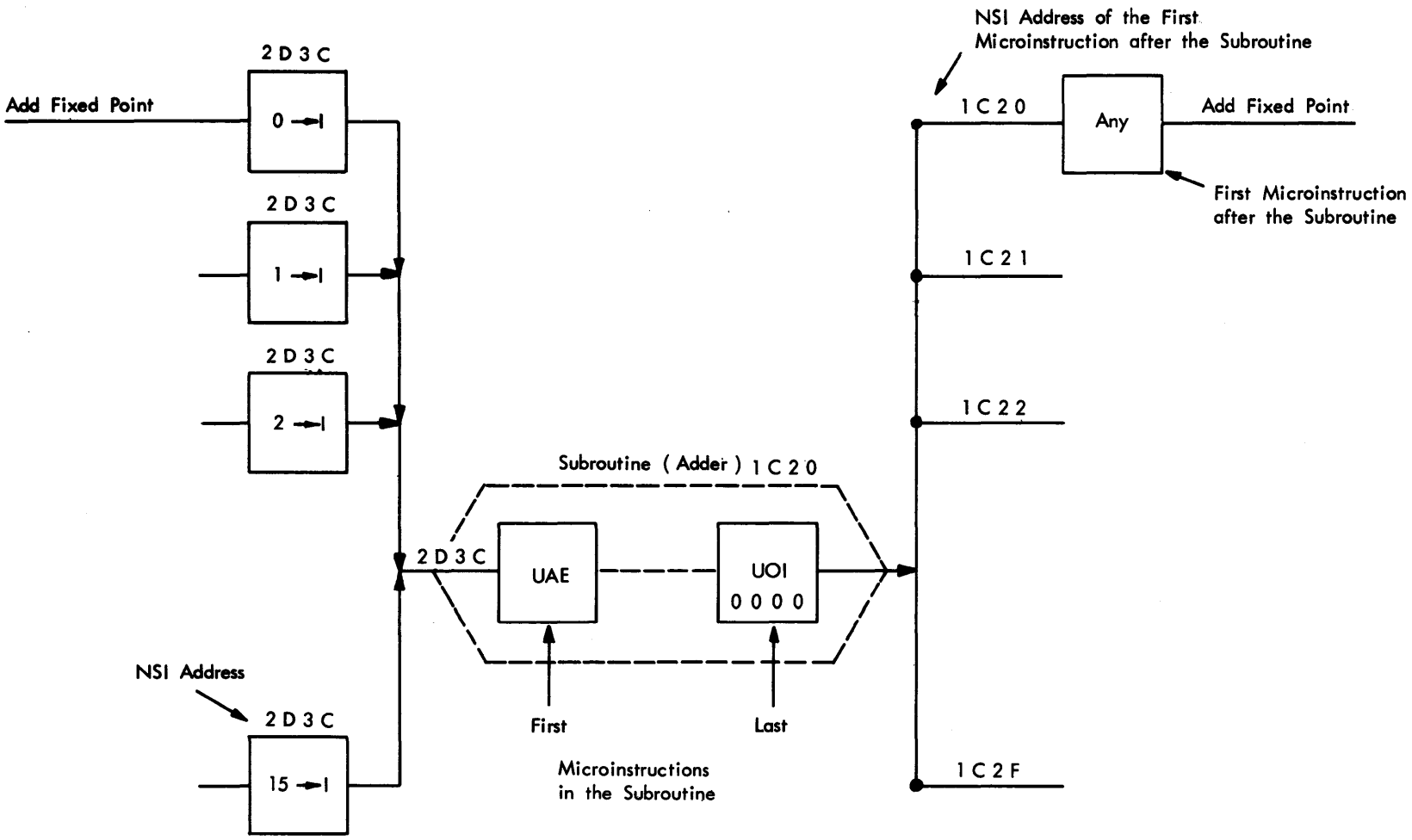


Figure 3-19 Entering and Leaving the Adder Subroutine

Example: add 3 and 5 = 8

Decremented Operand	Incremented Operand
3	5
2	6
1	7
0	8 Sum
15	
Carry	

A possible carry from a previous add operation is processed first. DR-R, in which one Operand is located, is then tested for an 8 bit. In order to save time (statistically) a determination is made to see if DR-R might contain a value less than 8 and if it does, it is statistically probable that it is faster to decrement DR-R (to 0) and increment DR-L to the result than to decrement DR-L (to 0) and increment DR-R to the result (later transferred to DR-L).

If DR-R contains no 8 bit, the content of DR-R is decremented and the content of DR-L is incremented. If DR-R contains an 8 bit, the content of DR-L is decremented and the content of DR-R is incremented.

Circuit Description (Figure 3-20))

When entering the Binary Adder, a test is made to check on a previous carry. This carry may be left over from the last time the Adder was used. It is represented by a 1 bit in DR-E.

It is assumed that there was no carry and therefore a decision must be made as to which register is to be decreased and which is to be increased. Normally DR-R is decreased while DR-L is increased until DR-R goes below zero and a carry occurs. The carry is the indication that the final result of the addition is in DR-L. In certain cases it is more economical to decrease DR-L and increase DR-R because it takes fewer increase/decrease operations to arrive at the final result. This is the case whenever the content of DR-R is bigger than the content of DR-L. Therefore a test is made for an 8 bit in DR-R.

If there is no 8-bit in DR-R, the normal mode of operation is used, that is R-1 and L + 1. As soon as an R-1 operation produces a carry, the final result is in DR-L and the Adder Routine is left.

If an 8-bit was found in DR-R, the opposite mode of operation is used, that is L-1 and R+1. As soon as an L-1 operation produces a carry, the final result is in DR-R. Since the final result must always be in DR-L, the content of DR-R is moved to DR-L before the Adder Routine is left. It is always the carry from a "minus one operation" that indicates the final result, regardless of the register in which it occurs.

When a carry occurs due to a "plus one operation", the partial result is in DR-L and a 1 bit is set into DR-E, indicating that a carry into the next higher order position is necessary.

This carry is accomplished the next time the Adder is used. When the 1-bit is present in DR-E, DR-L is increased by one.

When the L+1 operation again produces a carry, the content of DR-R is moved into DR-L and another carry into the next position is necessary.

When the L+1 operation results in no carry, the previous carry bit is reset because the carry was already taken into account. The routine is continued as before.

When operating in the normal way, that is R-1 and L+1, the possibility of a carry exists during the L+1 operation. After this operation the correct partial result is already in DR-R. Therefore the content of DR-R is moved into DR-L and then the 1-bit is set in DR-E to indicate that a carry into the next higher order position is necessary.

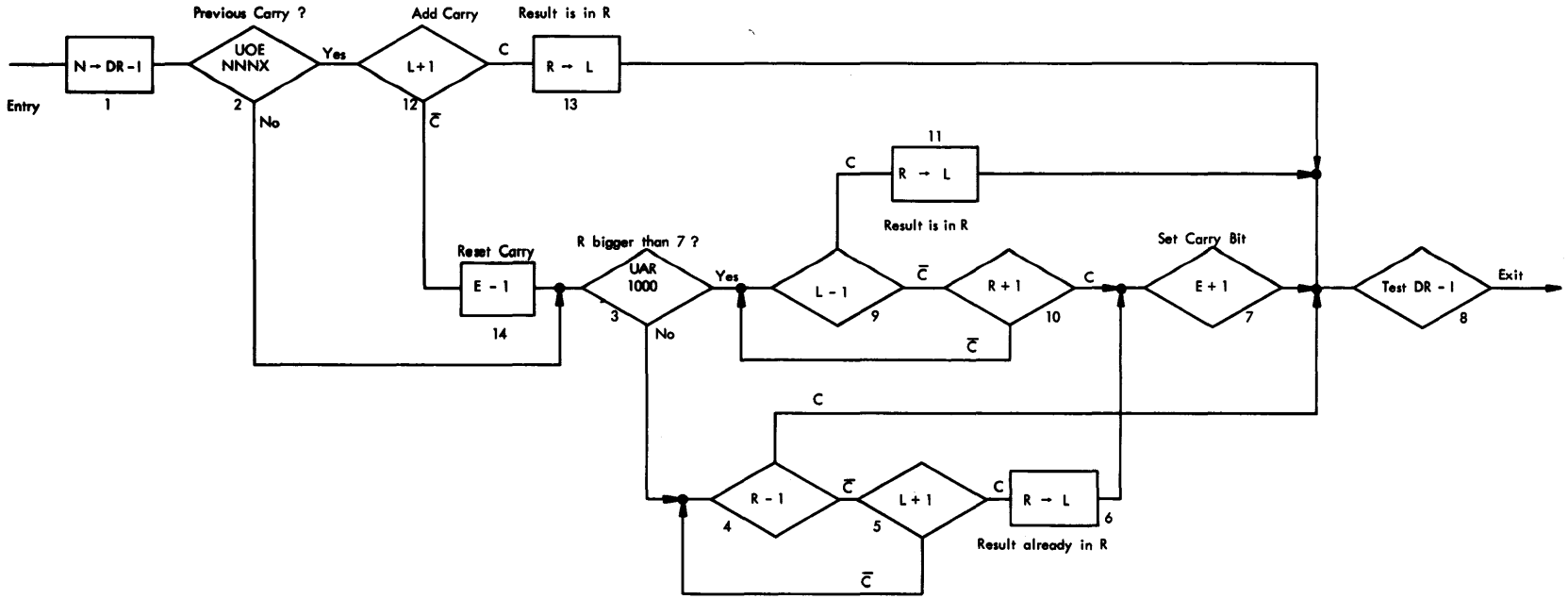


Figure 3-20 Adder Subroutine

Example 1. Carry + 3 + 15 = 3 and Carry

Step	Micro-instruction	DR-E	DR-R	DR-L
Entry		1	3	15
1	Test previous Carry			
2	L + 1	Yes		0 Carry
3	R to L			3
10	Exit	1 Carry to next position		3 = Sum

Example 2. 9 + 8 = 1 and Carry

Step	Micro-instruction	DR-E	DR-R	DR-L
Entry		0	9	8
1	Test previous carry	No		
5	Test DR-R for an 8 bit		Yes	
6	E + 1	1		
7	L - 1			7
9	R + 1		10	
7	L - 1			6
9	R + 1		11	
7	L - 1			5
9	R + 1		12	
7	L - 1			4
9	R + 1		13	
7	L - 1			3
9	R + 1		14	
7	L - 1			2
9	R + 1		15	
7	L - 1			1
9	R + 1		0 Carry	
10	Exit	1 Carry to next position		1 = Sum

Example 3. Carry + 9 + 5 = 15

Step	Micro-instruction	DR-E	DR-R	DR-L
Entry		1	9	5
1	Test previous Carry	Yes		
2	L + 1			6
4	E - 1	0		
5	Test DR-R for an 8 bit		Yes	
6	E + 1	1		
7	L - 1			5
9	R + 1		10	
7	L - 1			4
9	R + 1		11	
7	L - 1			3
9	R + 1		12	
7	L - 1			2
9	R + 1		13	
7	L - 1			1
9	R + 1		14	
7	L - 1			0
9	R + 1		15	
7	L - 1			15 Carry
8	E - 1	0		
9	R + 1		0 Carry	
10	Exit	0 No carry to next position		15 = Sum

Example 4. Carry + 7 + 6 = 14

Step	Micro-instruction	DR-E	DR-R	DR-L
Entry		1	7	6
1	Test previous Carry	Yes		
2	L + 1 (Add Carry)			7
4	E - 1 (Reset Carry)	0		
5	Test DR-R for an 8 bit		No	
11	R - 1		6	
12	L + 1			8
11	R - 1		5	
12	L + 1			9
11	R - 1		4	
12	L + 1			10
11	R - 1		3	
12	L + 1			11
11	R - 1		2	
12	L + 1			12
11	R - 1		1	
12	L + 1			13
11	L + 1		0	
12	L + 1			14
14	Exit	0 No Carry to next position		14 = Sum

Example 5. Carry + 7 + 9 = 1 and Carry

Step	Micro-instruction	DR-E	DR-R	DR-L
Entry		1	7	9
1	Test previous Carry	Yes		
2	L + 1			10
4	E - 1	0		
5	Test DR-R for an 8 bit		No	
11	R - 1		6	
12	L + 1			11
11	R - 1		5	
12	L + 1			12
11	R - 1		4	
12	L + 1			13
11	R - 1		3	
12	L + 1			14
11	R - 1		2	
12	L + 1			15
11	R - 1		1	
12	L + 1			0 Carry
13	E + 1	1		
11	R - 1		0	
12	L + 1			1
11	R - 1		15 Carry	
14	Exit	1 Carry to next position		1 = Sum

Compare Table with Two Result Exits

- Objective: Test the contents of a DR for equal (=) or unequal (≠) by comparing the contents of the DR with the Mask of a Use micro-instruction.
- There is one equal exit and fifteen unequal exits.
- When the table is entered via a UXX micro-instruction, the table is left by the equal exit if the DR contains the same bit pattern as the Mask. Otherwise the table is left via one of the unequal exits.
- When the table is entered via a UAX micro-instruction, the table is left by the equal exit if the DR and the Mask generates a NSI address with "zero" bits in the 4 low order positions (X XX X 00). Otherwise the table is left via one of the unequal exits.

Description (Figure 3-21)

NOTE: The heavy lines in Figure 3-21 show the micro-instructions of the Compare Table. All other micro-instructions shown are for entering the table.

A constant (0 through 15) is moved into DR-I to set up a value via which the table will be left whether the result exits via the equal or unequal exit.

This constant enables the micro-program to branch back into the proper main micro-program.

If the Compare table is entered via a Use micro-instruction and the last four bits of the modified USI address are "zero" (X XX X 00), a UI micro-instruction is addressed. This micro-instruction is modified by the contents of DR-I to return to the proper point in the main micro-program via the Equal exit of the Compare table. If the last four bits of the modified NSI are not zero (that is, a "one" in any position) one of 15 UI micro-instructions is addressed. The addressed UI micro-instruction is modified by the contents of DR-I to return to the proper point in the main-micro program via one of the 15 Unequal exits of the Compare Table.

Example 1. DR-E contains 0001. Following the compare operation, the main micro-program is resumed via the equal (=) exit 7 because DR-E contains 0001.

1. Move /7/ into DR-I to leave the table either by the equal or unequal exit 7.
2. The NSI address of the UAE micro instruction is given as 2 13 3 14 for purposes of this example.

The Mask (1110 = 14) of this micro-instruction is ANDed with the contents of DR-E. Because DR-E contains 0001, the NSI address is modified to 2 13 3 00. This is the address of the UI micro instruction which generates the NSI address to return, via the equal exit of the table, to the proper point in the main micro-program.

Example 2. DR-E contains 0101. Following the compare operation, the main-micro-program is resumed via the equal (=) exit 10.

3. Move /A/ into DR-I to leave the Table via exit 10 whether the compare is equal or unequal.
4. For purposes of this example, the NSI address of the UX1 micro-instruction is given as 2 13 3 05. The Mask (0101 = 05) of this micro-instruction is exclusive ORed with the contents of DR-E. Because DR-E contains 0101 (5) the NSI address is modified to 2 13 3 00. This is the address of the UI micro-instruction which generates the NSI address to return, via the equal exit of the table, to the proper point in the main micro-program.

Validity and Zero Test Table

- Objective: Three of the entries test the contents of a DR for one of three decimal values: /0/, /1/ through /9/, and /10/ through /15/.
- This subroutine has four entries.
- One entry tests for two decimal values /0/ through /9/ and /10/ through /15/.
- The table is used in decimal arithmetic.

Description (Figure 3-22)

1. Exit 0 if the tested DR contains /0/.
2. Exit 1-9 if the tested DR contains /1/ through /9/.
3. Exit 10-15 if the tested DR contains /10/ through /15/.

NOTE: The heavy lines in Figure 3-22 show the micro-instructions which are labeled Validity and zero Test table. All other micro-instructions are used to enter the table.

The subroutine has four entries to test the contents of DR-U, L and P for 0, 1-through 9, and 10 through 15.

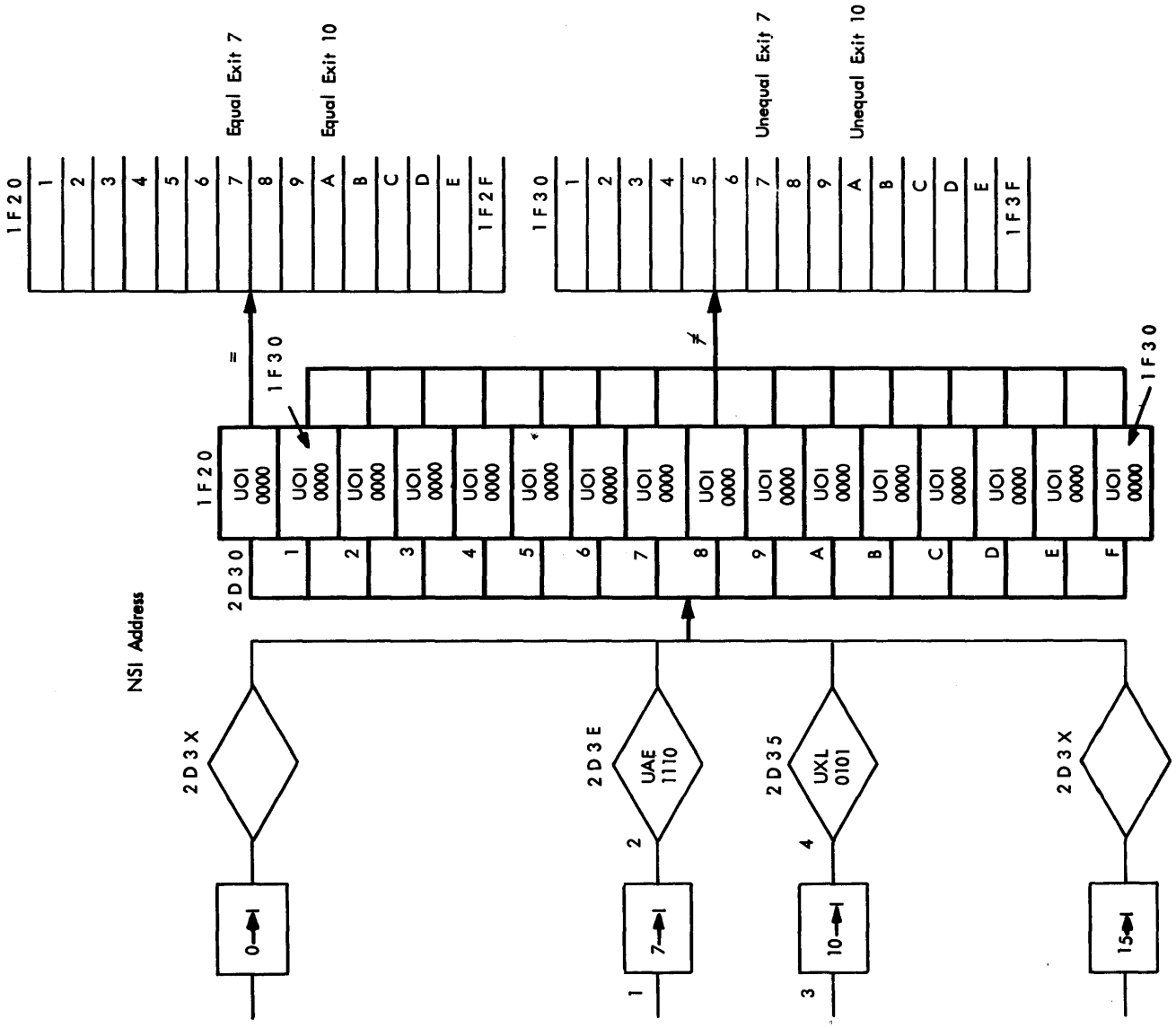


Figure 3-21. Compare Table with Two Exits

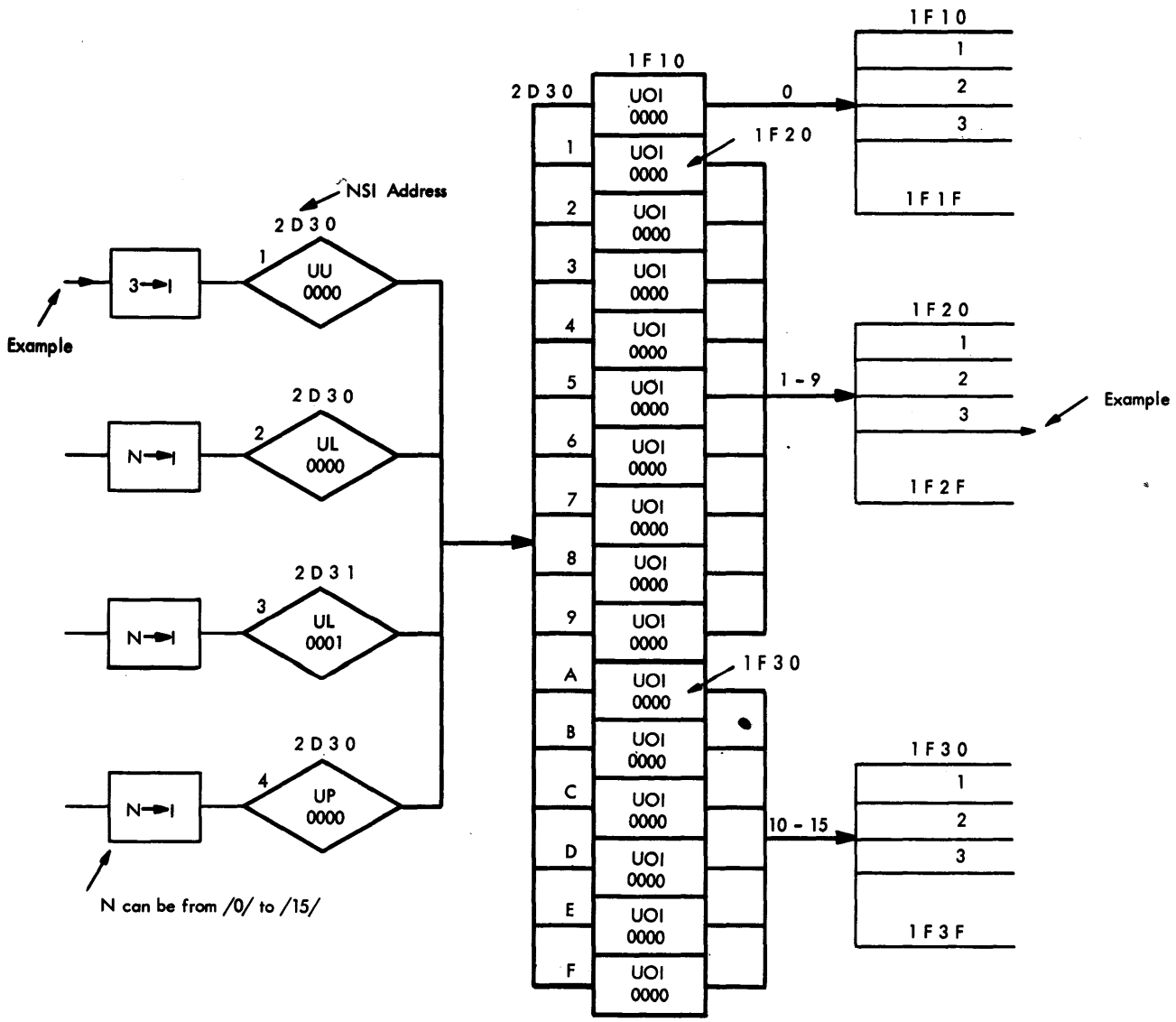


Figure 3-22 Validity and Zero Test Table Subroutine

NOTE: Entry 3 tests the content of DR-L for 0 through 9 and 10 through 15. If DR-L contains a /0/ the micro program exits the table by the exit 1-9.

Example: DR-U contains a /6/ = 0110. Following the Validity and Zero test operation the main micro-program is to be resumed via the 1-9 exit 3. The addresses of the 16 UI micro-instructions are 2 13 3 XX. Move /3/ into DR-I so that the table is left by exit 3 of either 0, 1 through 9, or 10 through 15.

For purposes of this example, the NSI address of the UU micro-instruction is given as 2 13 3 00. The Mask (0000) of this micro-instruction is ORed with the contents of DR-U. Because DR-U contains 0110 (6), the NSI address is modified to 2 13 3 06. Which is the address of the UI micro-instruction that (NSI address 1 15 2 00) generates the NSI address (1 15 2 03) to return, via the 1-9 exit 3 of the table, to the proper point in the main micro-program.

DR-S, T, R plus 1 (STR+1)

- Objective: Modify plus 1 the contents (address) of DR-(A) S, T, and R.

Description (Figure 3-23)

This subroutine is entered via one of two paths: Entry A tests to see if an I/O Service phase is required. Entry B does not test for I/O Service required.

1. The contents of DR-R is incremented by 1. If no carry occurs, the contents of DR-I is used to branch back to the main micro-program. For a carry see step 2.
2. The contents of DR-T is incremented by 1 to propagate the carry caused by incrementing DR-R. If no carry occurs when DR-T is incremented by 1, the contents of DR-I is used to branch back to the main micro-program. For a carry (from T + 1) see step 3.
3. The contents of DR-S is incremented by 1 to propagate the carry caused by incrementing DR-T. If no carry occurs when DR-T is incremented by 1, the contents of DR-I is used to branch back to the main micro-program. For a carry (from S + 1) see step 4.

4a. If the CPU has only 4K storage, a carry caused by incrementing DR-S is detected as a wrap around condition. If wrap around occurs a /3/ is moved into DR-I to indicate an addressing error. The CPU is stopped by micro-program routine Error Stop (Figure 3-27a).

4b. If the CPU has an 8K storage, a carry caused by incrementing DR-S increments (turns on) the 1 bit in DR-A. If the CPU has a 16K storage, the 8K latch in DR-A is turned on. The DR-A cannot be turned on directly by the carry. Before DR-A is turned on, a micro-instruction (Move S to L) moves the contents (on/off condition) of DR-A into DR-U so that the on/off condition of DR-A can be tested.

5. Test DR-U for a 1 bit. If DR-U (DR-A) does not contain a 1 bit, continue as shown in step 6. If DR-U contains a 2 bit, the 8K latch is turned on. If DR-U (DR-A) contains a 1 bit, continue as shown in step 8.
6. Move /1/ into DR-U to initiate the carry (Item 4b) into DR-A.
7. Move the contents of DR-L back into DR-S. This micro-instruction moves the /1/ in DR-U into DR-A(4K latch is turned on). In the next micro-instruction, the contents of DR-I is used to branch back to the main micro-program.
8. If the CPU has an 8K storage, and if the DRA 1 bit is on, a carry caused by increasing DR-S is detected as a wrap around condition. If wrap around occurs, a /3/ is moved into DR-I to indicate this as an error condition. The CPU is stopped by the micro-program routine Error Stop (Figure 3-27a).

STR-minus 1(STR-1)

- The contents (address) of DR-(4K), S, T, and R are modified by minus 1 by this subroutine.

Description (Figure 3-24)

This subroutine is entered via two paths: Entry A tests for I/O Service phase. Entry B does not test for I/O Service phase.

1. The contents of DR-R is decremented by 1. If no carry occurs, use the content of DR-I to branch back to the main micro-program. For a carry, see step 2.
2. The contents of DR-T is decremented by 1. If no carry occurs, use the contents of DR-I to branch back to the main micro-program. For a carry, see step 3.
3. The contents of DR-S is decremented by 1. If no carry occurs, use the contents of DR-I to branch back to the main micro-program. For a carry, see step 4.

NOTE: Steps 4, 5, and 6 modify the contents of DR-A from 1 to 0 because of the carry produced by decrementing DR-S.

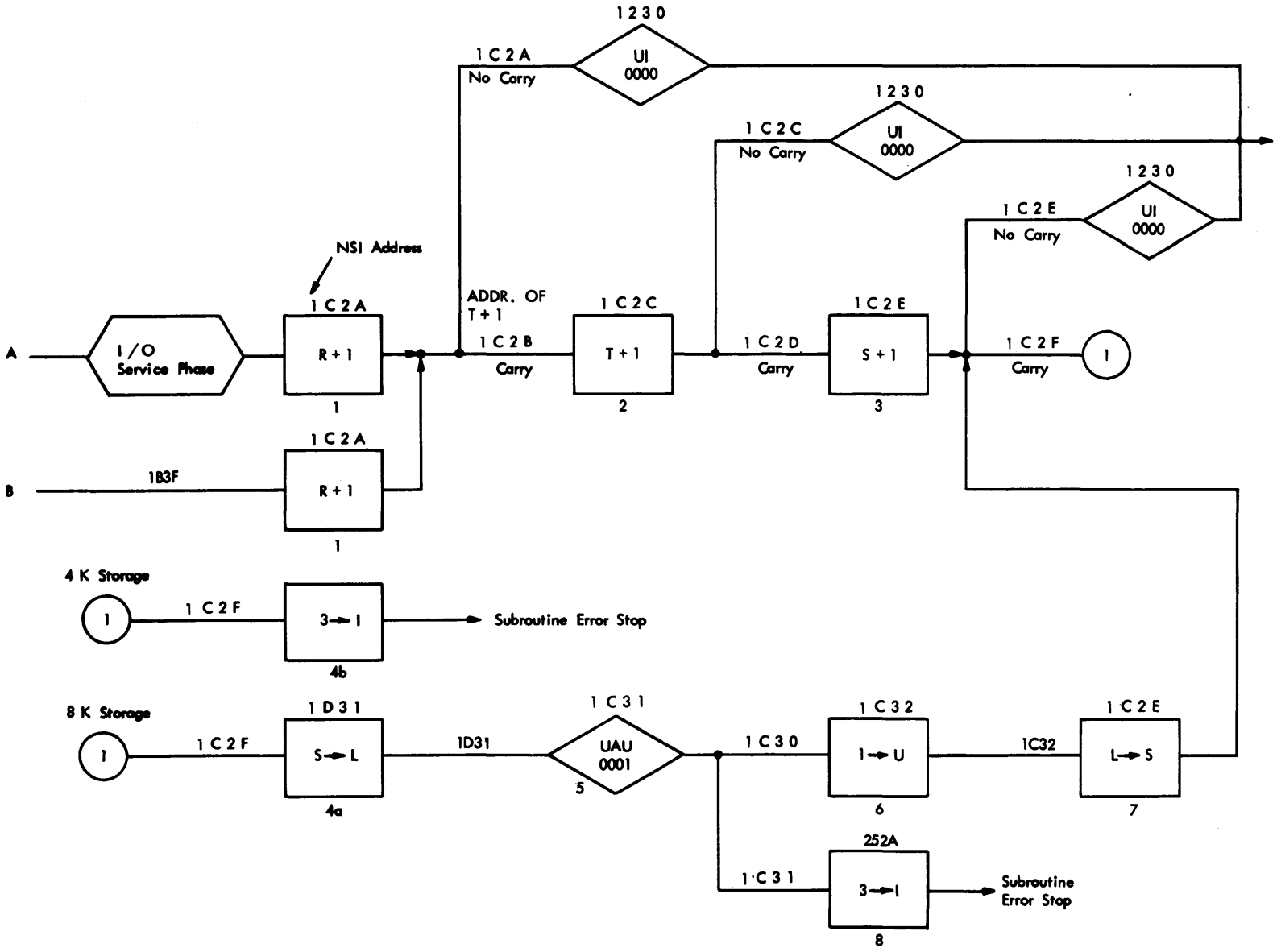


Figure 3-23 STR + 1 Subroutine

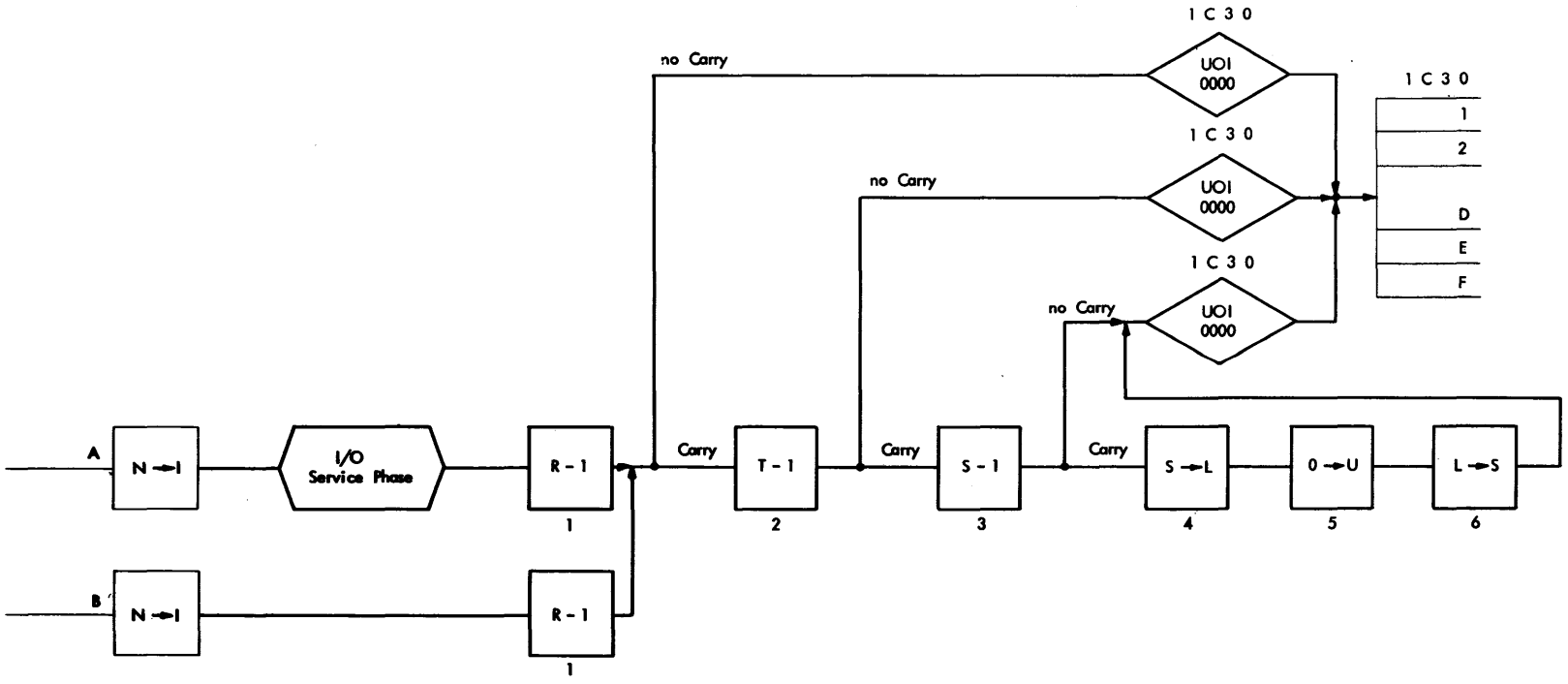


Figure 3-24 STR-1 Subroutine

4. Save the contents of DR-S by moving it into DR-L. The DR-A can only be set with a Move L to S, and the contents of DR-S must not be destroyed.
5. Move /0/ into DR-U, because the 4K latch is reset to /0/ if DR-U contains /0/ during the next step.
6. The contents of DR-L is returned to DR-S by the Move L to S micro-instruction. DR-A is set to /0/ because DR-U contains no 1 bit during the Move L to S operation. See Main Storage Addressing for 8K. Use the contents of DR-I to branch back to the main micro-program.

Compare Table with Three Result Exits

- This operation compares the contents of DR-X (X = E,S,T,R, or U) with the contents of DR-L.

- The results of the comparison are:
  Less than (<) the contents of DR-L
  Greater than (>) the contents of DR-L

Description (Figure 3-25)

NOTE: The heavy lines in Figure 205 show the micro-instructions for the Compare Table. All other micro-instructions shown are for entering the table. All address but not the Masks are for the examples only.

The table is entered by testing the contents of DR-X (X can be E,S,T,R, or U) with a UOX micro-instruction to address one of 16 UXL micro-instructions. The Mask of the selected UXL has the same bit pattern as the contents of DR-X. The Mask of the selected UXL is Exclusive ORed with the contents of DR-L to generate one of 16 possible NSI addresses.

Equal Results. If the generated NSI address is /6D30/, the Mask (contents of DR-X) and the contents of DR-L are equal because a /0/ in the low order part of the NSI address can be generated only by an Exclusive OR when the Mask and the DR have the same bit pattern. The NSI address /6D30/ selects a USI micro-instruction which is modified by the contents of DR-I to return to the proper point in the main micro-program by way of the equal (=) exit.

Not Equal Result. If the generated NSI address is /6D31/ through /6D3F/ (15 addresses), the compare is not equal and the contents of DR-L is tested by a UAL for further comparing (< or >). The 15 UAL have four different types of Masks /1/, /2/, /4/, or /8/. The type of Mask selected depends upon the high order bit of DR-X or DR-L.

Example: Contents of DR-X = 0101
Contents of DR-L = 0011
The high order bit of both is the 4 bit in DR-X. Therefore, the selected UAL has a Mask /4/.

The UAL ANDs the Mask (/1/, /2/, /4/, or /8/) which contains the high order bit of DR-L or DR-X.

Low Result. If the high order bit is set from DR-X, the contents of DR-L is lower than the contents of DR-X. This condition (<) is detected by the generation of a NSI address with a /0/ in the low order position (NSI address is /6F20/). This address selects a UOI which is modified by the contents of DR-I to return to the proper point in the main micro-program by way of the lower than (<) exit of the table.

High Result. If the high order bit is set from DR-L, the contents of DR-L is greater than (>) the contents of DR-X. This (>) condition is detected by ANDing the high order bit of DR-L (which is represented in the Mask of the UAL) with the contents of DR-L to generate a NSI address. The NSI address contains a /1/, /2/, /4/, or /8/ in the low-order position. All four NSI addresses are selecting a UOI micro-instruction which is modified by the contents of DR-I to return to the proper point in the main micro-program by way of the greater than (>) exit. Three examples follow to illustrate Equal, Less Than, and Greater Than conditions:

1. Equal Example: DR-L = /6/, DR-E = /6/
   a. Move /2/ into DR-I to leave the table via the exit 2.
   b. The UOE has the punched NSI address /6E20/. The ORing of the Mask /0/ and the /6/ in DR-L generates the NSI address /6E26/ which selects a UXL.
   c. The selected UXL has the punched NSI address /6E36/. The Exclusive ORing of the Mask /6/ and the /6/ in DR-E generates the NSI address /6E30/ which selects a UOI.
   d. The selected UOI has the punched NSI address /2430/. The ORing of the Mask /0/ and the /2/ in DR-I gives the NSI address /2432/ which is one of the equal (=) exits of the table.
2. Less Than Example: DR-L = /7/, DR-S = /C/(12)
   a. Move 0 into DR-I to leave the table by way of exit 0.
   b. The UOS has the punched NSI address /6E20/. The ORing of the Mask /0/ and the /C/ in DR-S generates the NSI address /6E2C/ which selects a UXL.

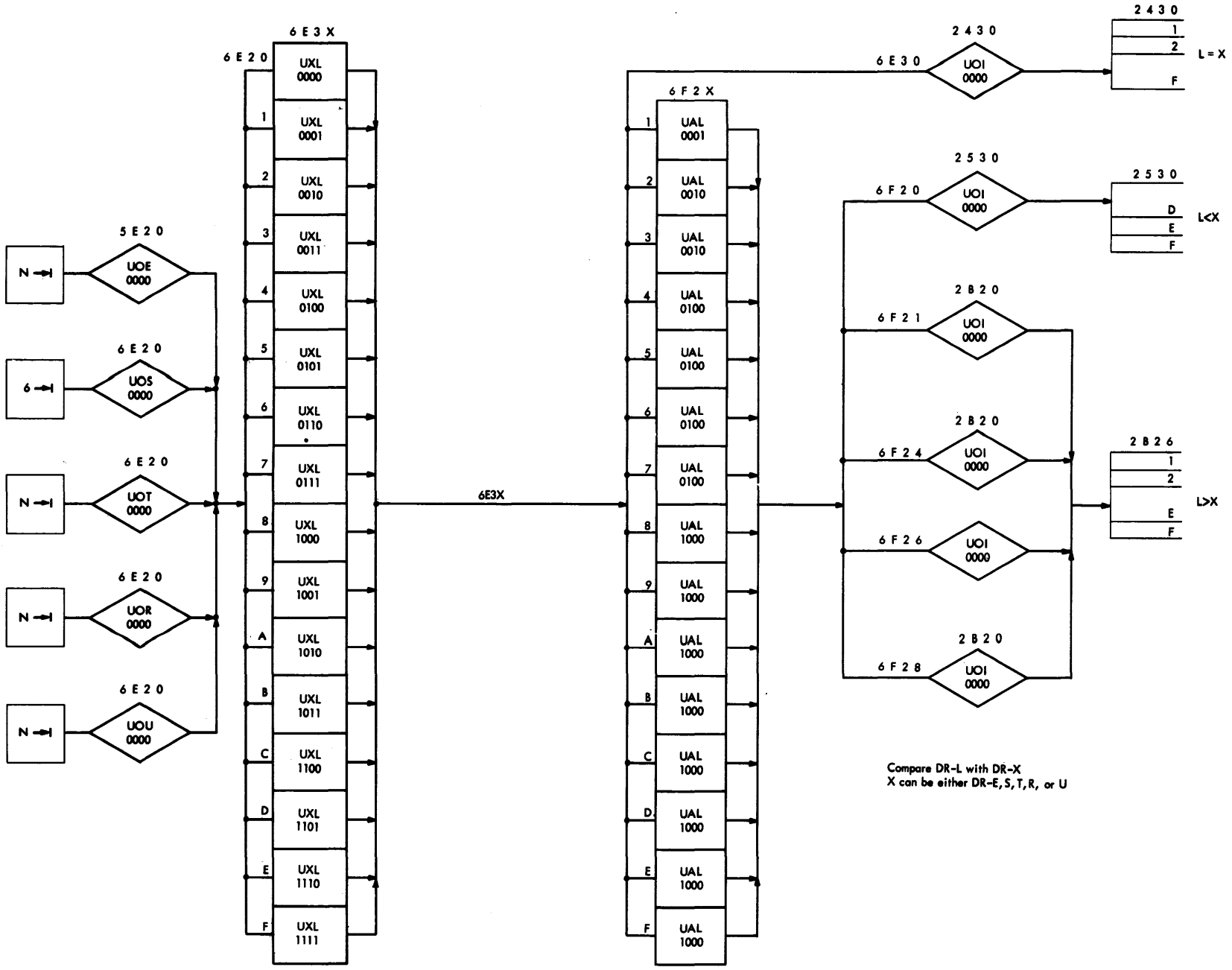


Figure 3-25 Compare Table with Three Exits Subroutine

c. The selected UXL has the punched NSI address /6E3C/. The Exclusive ORing of the Mask /C/ and the /7/ in DR-L generates the NSI address /6E3B/ which selects a UAL. The UAL has a punched NSI address /6F28/. The ANDing of the Mask /8/ and the /7/ in DR-L generates the NSI address /6720/ which selects a UOI.

d. The selected UOI has the punched NSI address /2530/. The ORing of the Mask /0/ and the /0/ in DR-I gives the NSI address /2530/ which is one of the less than (<) exits of the Table.

3. Greater Example DR-L = /9/, DR-U = /0/

a. Move /E/ into DR-I to leave the Table by way of exit E.

b. The UOU has the punched NSI address /6E20/. The ORing of the Mask /0/ and the /0/ in DR-U generates the NSI address /6E20/ which selects a UXL.

c. The selected UXL has the punched NSI address /6E30/. The Exclusive ORing of the Mask /0/ and the /7/ in DR-L generates the NSI address /6E37/ which selects a UAL.

d. The UAL has a punched NSI address /6F24/. The ANDing of the Mask /4/ and the /7/ in DR-L generates the NSI address /6F24/ which selects a UOI.

e. The selected UOI has the punched NSI address /2B20/. The ORing of the Mask /0/ and the /E/ in DR-I gives the NSI address /2B2E/ which is one of the greater (>) exits of the table.

Changing the Contents of DR-L

- The Identity table is used to change the contents of DR-L.

- Changing the contents of DR-L can be done for any of the following purposes:
  Inserting one or more bits.
  Removing one or more bits.
  Receiving the result of ORing DR-R and DR-L.
  Receiving the result of ANDing DR-R and DR-L.
  Receiving the complement value of DR-L.

Description (Figure 3-26)

Inserting Bits. To insert one or more bits into DR-L, the Identity table is entered by means of the OR table. The OR table consists of 16 UOL with 16 different Masks /1/ through /F/. To insert a certain bit pattern (8) into DR-L, a UOI with the same bit pattern as Mask (8) must be addressed. The UOL ORs the Mask (inserted bit pattern) with the contents of DR-L to generate a NSI address which selects an N to L micro-instruction in the Identity table. The N value in the selected micro-instruction represents the result of the ORing (DR-L plus inserted bits).

The N to L operation moves the new value into DR-L and the table is left by means of a UOI to return to the proper point in the main micro-program.

Removing Bits. To remove one or more bits from DR-L, the Identity table is entered by means of the AND table. The AND table consists of 16 UAL with 16 different Masks (/1/ through /F/). To remove a certain bit (8) from DR-L, a UAL with a zero bit in the same bit position (8) of the Mask, must be addressed. The UAL ANDs the Mask (zero bits in remove position) with the contents of DR-L to generate a NSI address which selects an N to L micro-instruction in the Identity table. The N value in the selected micro-instruction represents the result of the ANDing. The N to L operation moves the result into DR-L. The table is left by means of a UOI to return to the proper point in the main micro-program.

Receiving the Result of ORing DR-R and DR-L. To receive the result of ORing DR-R and DR-L, the OR table is entered by means of a UOR with Mask /0/. The ORing of the Mask /0/ and the contents of DR-R selects a UOL in the OR table. The selected UOL has the same bit pattern in the mask as the contents of DR-R. The ORing of the Mask (same bit pattern as DR-R) and the contents of DR-L generates a NSI address which selects an N to L instruction in the Identity table. The N value in the selected N to L micro-instruction represents the result of ORing DR-R and DR-L. The N to L operation moves the result into DR-L and the table is left by means of a UOI to return to the proper point in the main micro-program.

Receiving the Result of ANDing DR-R and DR-L. To receive the result of ANDing DR-R and DR-L, the AND table is entered by means of a UOR with Mask /0/. The ORing of the Mask /0/ with the contents of DR-R selects a UAL in the AND table. The selected UAL has the same bit pattern in the Mask as the contents of DR-R. The ANDing of the Mask (same bit pattern as DR-R) and the contents of DR-L generates a NSI address which selects a N to L micro-instruction in the Identity table. The N value in the selected N to L micro-instruction represents the result of ANDing DR-R and DR-L. The N to L operation moves the

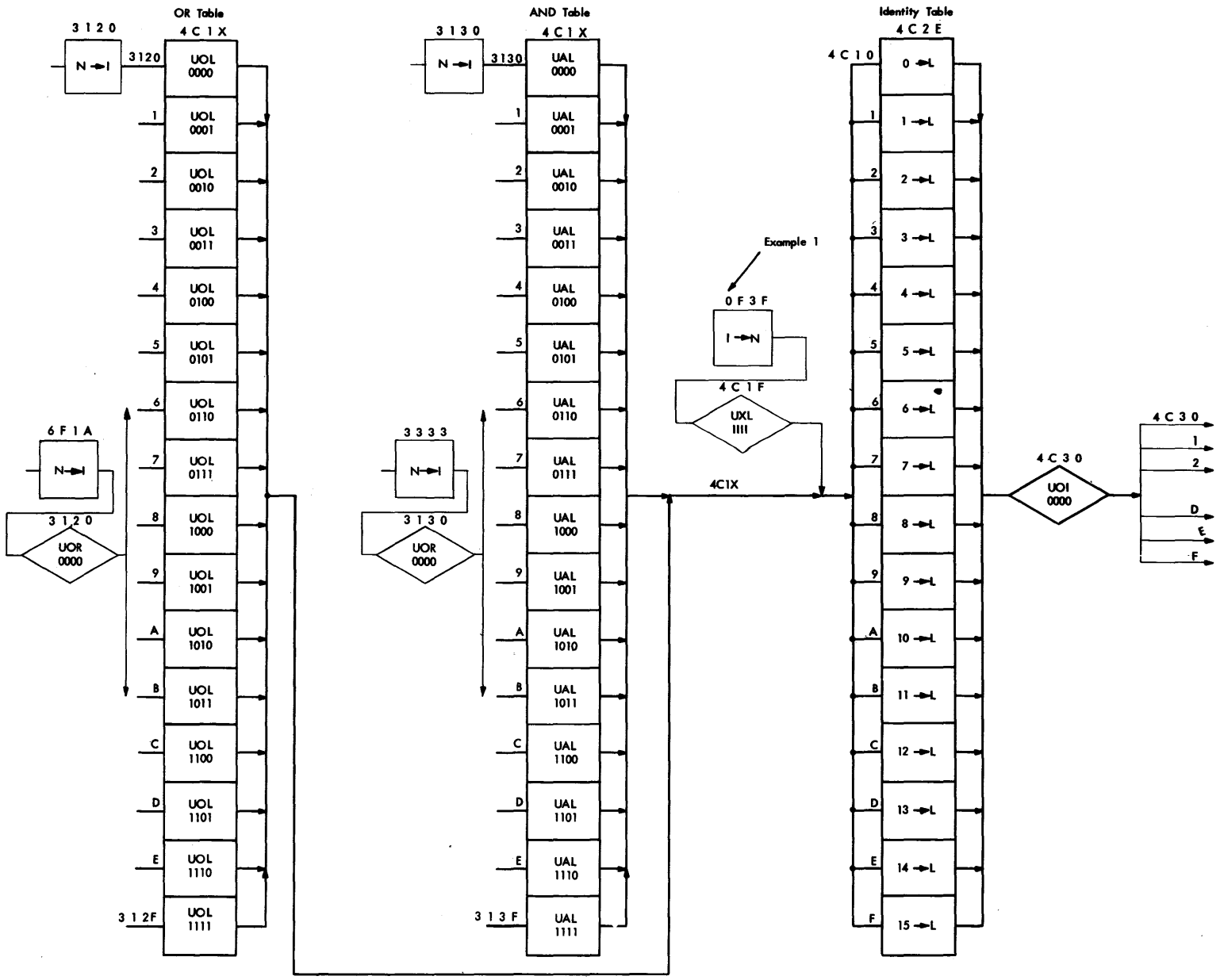


Figure 3-26 OR-AND-Identity Table

result into DR-L and the table is left by means of a UOI to return to the proper point in the main micro-program.

Complementing the Contents of DR-L. To complement the contents of DR-L (Arithmetic operations) the Identity table is entered by means of a UXL with Mask /F/ (1111). The Exclusive ORing of the Mask (F) with the contents of DR-L generates a NSI address which selects an N to L micro-instruction in the Identity table. The N value in the selected N to L micro-instruction represents the complement value of DR-L. (See the following example. The N to L operation moves the result into DR-L and the table is left by means of a UOI to return to the proper point in the main micro-program.

Example: Complementing of /6/ (0110) in DR-L = /9/ (1001).

1. A /3/ is moved into DR-I to return to the proper point in the main micro-program. The NSI is located in /073F/.
2. The UXL in location /0737/ has a punched NSI address /4C1F/. The Mask /F/ (1111) is Exclusive ORed with the /6/ (0110) in DR-L. The modified NSI address is /4C19/ because the Exclusive ORing of the Mask = 1111 and the contents of DR-L = 0110 gives a 1001 (/9/) in the low order position of the NSI address. The NSI address /4C19/ selects the 9 to L micro-instruction in the Identity table. After the 9 to L operation, DR-L contains the complemented value.
3. The NSI address in the 9 to L micro-instruction is /4C2E/ which selects a UOI with the punched NSI address /4C30/. The ORing of the Mask /0/ and the /2/ in the DR-I generates the NSI address /4C32/ to return to the proper point in the main micro-program.

MANUAL OPERATIONS

Figure 3-27 shows the relation of the manual operations with the I-phase.

SYSTEM RESET KEY

- Objectives: Reset the system when an error is present and prepare the system to start the CPU.
- Generate a System Reset signal which places the latches shown in Table 3-1 in their initial or reset condition.
- Generate the Reset Condition signal which places the latches shown in Table 3-2 in their initial or reset condition.
- Initiate the Start Reset micro-program loop (Start Address 3 F 3 F is forced).
- All DRs are cleared.

Description (Figure 3-27a)

The System Reset Key must be pressed whenever:

1. A Power failure occured.
2. The main line switch was turned off.
3. The Emergency Power Off switch was pulled.
4. An error is present.

In all these cases it is apparent, that something went wrong. It would not suffice to simply restore power and continue working because normal operation of the CPU could not be expected. (Bits may have been lost, and so on). Therefore, the System Reset Key must prepare the CPU for an initial start.

When the System Reset Key is pressed, the following operations take place:

a. ROAR latches 10-21 are turned on, ROAR E latch is turned off. This enables the ROAR to read out the Start Address of the micro-program. The micro-program then performs the actual System Reset Routine.
b. To insure proper ROAR-Timing the following Ring latches are set or reset:
   a. Ring latches A1, C1 are turned on.
   b. Ring latches A2, A3, C2, C3 are turned off.

   This is necessary, because the Ring may have been set differently when the error or fault occured.
c. The Pre-latches 1, 2 and 3 are turned off to produce an Even ROS Address condition. This is necessary because the micro-program Start Address (3 F 3 F) would otherwise cause a ROS-Check.
d. The Reset Condition signal (Produced by the System Reset key) resets the following CPU Check latches: Process Check, Modifier Check, AHR Check 1, ROAR Check, STAR Check, Storage Check, Ring Check, ROS Check and Gate Decoder Check. Also Auxiliary latches 1 and 2 are reset. The reset of these latches is necessary because any one of these checks may have caused the error stop. The micro-program resets the system in the following manner (Figure

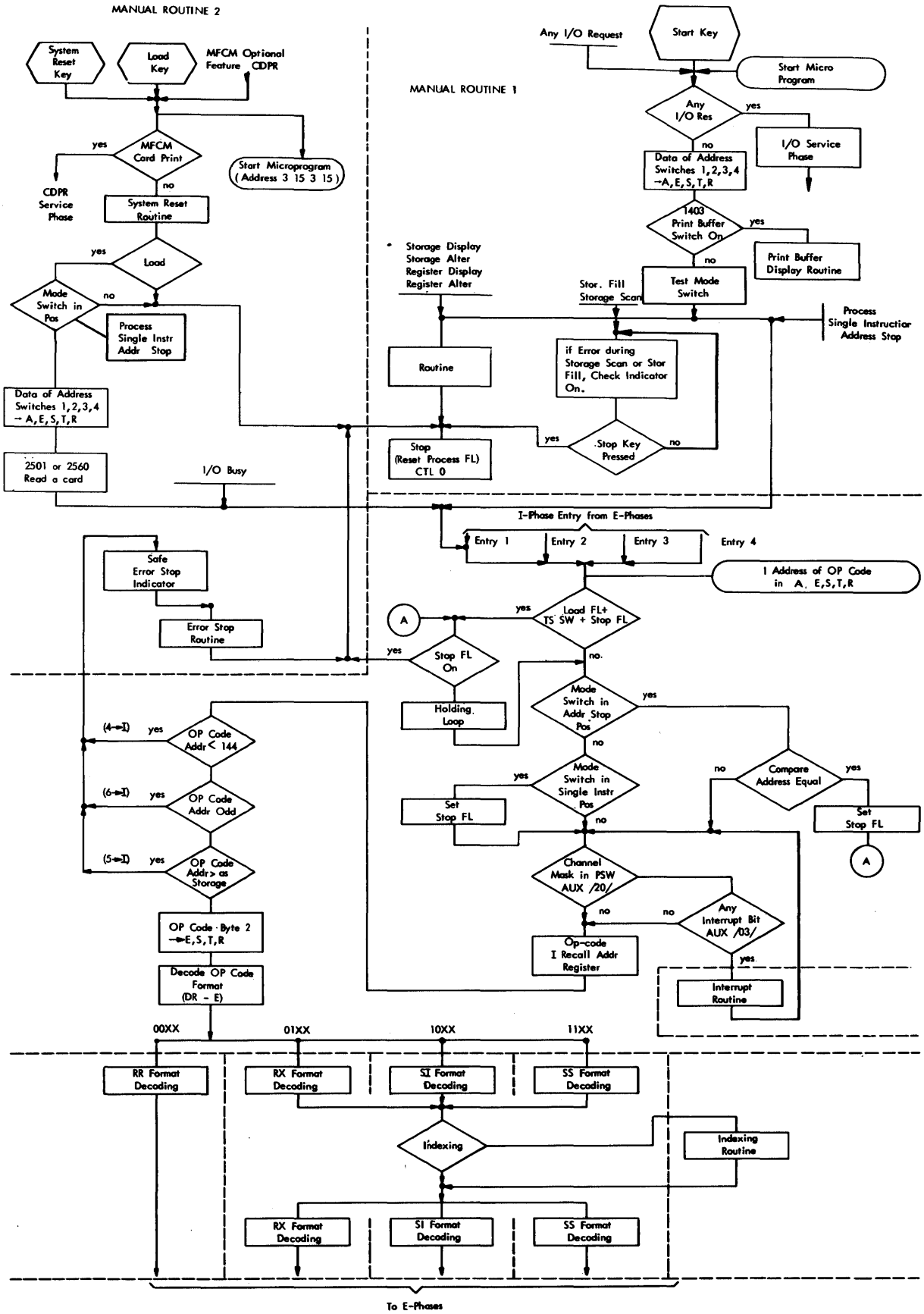


Figure 3-27 I-phase and Manual Routine Principle

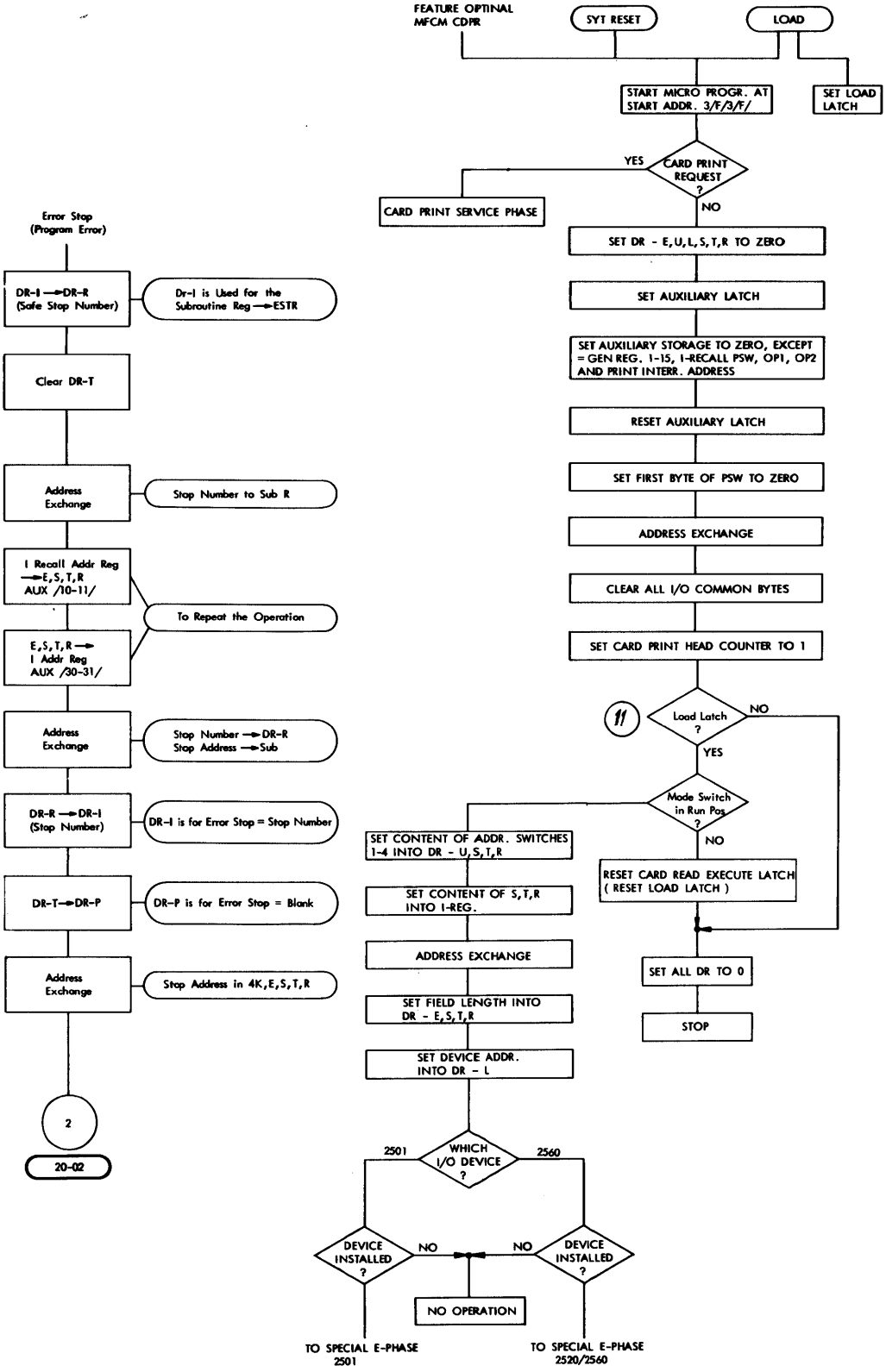


Figure 3-27a Manual Routine 2

2. The micro-instruction, which is read out with the Start Address is a Sense 1. The Sense 1 instruction checks for a card print request (After an error or power failure there would be none).
3. Next, the DR-E, U, L, S, T, R are set to Zero because these registers are needed in performing the ResetRoutine.
4. The Auxiliary latch 2 is turned on to enable a Fetch STR instruction to address Auxiliary Storage. (Normally a F STR addresses the Main Storage).
5. The bytes in the Auxiliary Storage are filled with zeros except the General Registers, the I-Recall Register, the Operand 1 and 2 Address Registers. The Program Status Word (PSW) as well as the Print interrupt Address are also left unchanged. These Registers contain important data of the customers program. (The customer may want to know where his program stopped).
6. Auxiliary latch 2 is turned off because it is not needed for further operations in the System Reset Routine
7. The first byte of the PSW is read out and zeros are read in to remove any previous condition code, channel mask, and ASCII Mode.
8. A Control 15 Instruction causes an address exchange to remove any previous addresses in the Sub-registers.
9. All I/O Common bytes (Which contained zeros and parity bits) are read out. This means that they are cleared completely.
10. The Print Head counter byte is set to zero. The zero sets the print head counter to its initial state. The System Reset Routine is identical with the Load Routine up to this point. They share a common loop.
11. A test now is performed to detect whether the Load key or the System Reset key was pressed.
12. If the Load key was pressed, the Mode switch is tested to see whether it is in the Run position.
13. If the Mode switch was not in the correct position, it is assumed that the Load key was pressed erroneously. Therefore the Read Execute latch and the Load latch are turned off and the DRs are cleared as in step 13.
14. If the System Reset key was pressed, DR-U, L, S, E, T, R, P and I are set to zero once more, because some of these registers still contain the addresses of their Sub-registers. The CPU then stops and this terminates the Reset Routine.
15. If the Mode switch was set correctly, the Load Routine proceeds.

The System Reset signal is generated by the following:

1. Power On Reset FL (ANDed with Start)
2. System Reset Key
3. Load Key
4. Card Print Cycle FL

The System Reset signal resets or sets the latches as shown in Table 3-1.

Reset Condition signal is generated by the following:

1. Power on FL
2. System Reset Key
3. Load Key
4. ROS Recycle FL (CE)

The reset Condition resets or sets the latches as shown in Table 3-2.

START-STOP CIRCUIT SEQUENCE

- When the Start key is pressed, the micro-program starts.
- The Start key turns on the Process latch.
- The Process latch is turned off by a CTL O Stop or a Process Check Stop.

Description (Figure 3-28)

Start Key On Sequence. The following sequence occurs when the Start Key is pressed. Note that the numbered paragraphs below refer to the numbered signal lines on Figure 3-28.

1-3 Pulses A, B, and C are continuous if power is on.

5 When the Start key is pressed, the Start latch is turned on with the next Pulse C.

6. The Start latch ANDs with Pulse A to turn on the Start Interlock latch. The Start Interlock latch prevents multiple start signals to the CPU when the Start key is held down. As the micro-program runs, the Start Interlock latch is turned off when the Start Key is released.

14. The Start Interlock latch ANDs with Pulse A to turn on the Force ROS latch for later use.

7. The Start Interlock latch ANDs with Pulse B to turn on the Delta Process latch. The Delta Process latch is used to collect the different stop conditions and to control the Process latch.

8. The Delta Process latch ANDs with Pulse A to turn on the Process latch.

9. The Delta Process latch ANDs with Start latch off and a time signal (0-250). The output of the AND switch is called Clock A and is active for the same time as Pulse A.

10. The Process latch output is ANDed with a time signal (400-550). The output of the AND switch is called Clock C and is active for the same time as Pulse C.

11. The ROAR Advance signal is active when no Fetch or Store micro-instruction is decoded.

12. Clock A is ANDed with the ROAR Advance signal to set the ROAR with the NSI address in the ROSDR. The ROSDR contains the CTL 0 micro-instruction which stopped the CPU after the previous operation was finished.

13. Clock A activates the TROS Timing circuit. This circuit gives two signals, the Driver Strobe (300-520) and a time signal (300-400). The Driver Strobe activates TROS and reads out the addressed micro-instruction into ROSDR.

Table 3-1. Latches Set or Reset by System Reset Signal

CPU		
ROAR 10-21 FLs	on	Start Address
ROAR E FL	off	3 15 3 15
Ring A1 FL	on	
Ring A2 FL	off	
Ring A3 FL	off	
Ring C1 FL	on	
Ring C2 FL	off	
Ring C3 FL	off	
Pre FL 1	on	
Pre FL 2	on	Even ROS Address
Pre FL 3	on	
ROS A ddr Equal FL (CE)	off	

Table 3-2. Latches Set or Reset by Reset Condition Signal

CPU		
Process Check FL	off	sample check FL
Modifier Check FL	off	
AHR Check FL1	off	
ROAR Check FL	off	
STAR Check FL	off	different CPU checks
Store Check FL	off	
Ring Check FL	off	
Gate Dec Check FL	off	
ROS Check FL	off	
AUX FL1	off	
AUX FL2	off	

15. The time signal (300-400), generated in the TROS Timing signal, is ANDed with the Force ROS latch to develop the Reset Strobe signal. The Reset Strobe resets the ROSDR at 300-400 time.
16. The empty ROSDR can now receive the micro-instruction which is read out by the Driver Strobe. The micro-instruction in ROSDR is decoded and executed.

As long as the Process latch is on, the TROS timing circuit is activated with each Clock A and the micro-instructions are consecutively processed.

Figure 3-28 illustrates how the Process latch is turned off by a CTL 0 or by a Process Check.

NOTE: The Process latch can also be turned off if the CE Micro-step switch is on, or if the CE Immediate Error Stop switch is on and an I/O error occurs.

Process Latch Off Sequence with CTL 0 Stop (Figure 3-28). When a CTL 0 micro-instruction in the ROSDR is decoded, the Process latch is turned off and the micro-program processing is interrupted (Stop) as follows:

1. The Force ROS latch is turned off with Pulse A to prevent the reset of the ROSDR (CTL 0 micro-instruction).

   NOTE: The NSI, after the CTL 0, is read out of the TROS but is not set into the ROSDR.

2. The Delta Process latch is turned off with CTL 0 and Pulse B to prepare for the reset of the Process latch.
3. Delta Process latch off prevents the generation of Clock A.
4. Pulse A and Delta Process latch off turns off the Process latch.
5. Process latch off prevents the generation of Clock C.

No Clock A and no Clock C prevent the processing of micro-instructions (CPU Stop).

Process Latch Off Sequence with Process Check Stop. When a Process check is detected (One of the 8 CPU checks occurs), the Process Check latch is turned on at Clock C time. The Process Check latch then causes the Process latch to go off.

1. Process Check latch on prevents the generation of Clock A.
2. Process Check latch on ANDs with Pulse A to turn off the Delta Process latch and the ROAR Advance latch.
3. The same Pulse A turns off the Process latch when the Delta Process latch is off.
4. Delta Process latch off prevents the generation of Clock A.
5. Process latch off prevents the generation of Clock C.
6. The lack of Clock A and Clock C prevents processing of micro-instruction (CPU stop).

   NOTE: During the cycle in which the Process check is detected, a micro-instruction is read out of the TROS and is set into ROSDR. ROSDR contains the micro-instructions following the erroneous micro-instruction.

STOP KEY

- Objective: Stop the program in process if the Mode switch is in Process or Storage Scan position.
- With the Stop Key pressed the CPU stops after the machine instruction in progress is completed. All time-shared I/O operations in process continue to completion.
- The Stop Key turns on the Stop Condition latch.
- The Stop-Condition latch is sensed (SENSE 15) at the beginning of the next I-phase.
- Stop Condition latch and Sense 15 activates Bus 2 line and turns on DR-L 2 bit.
- DR-L 2 bit is tested, and directs the micro-program to the micro-instruction CTL 0 which resets the Process latch.

Description (Figure 3-29)

At the beginning of each I-phase the Stop Condition FL, which can be turned on by the Stop Key, is tested. If the Stop Condition FL is on, the micro-program is directed to a CTL 0 operation which resets the Process FL.

In Figure 3-29, I-phase is entered either in Entry 1, 2, 3, or 4.

45. A test is made to determine amongst other things if the Stop Condition FL is on. Pressing the Stop key turns on the Stop Condition FL. With the Stop Condition FL on, proceed to step 59.
59. Store E, S, T, R in the I Address Register (AUX storage /30-31) using the ESTR - Reg Subroutine. DR-E, S, T, R contains the

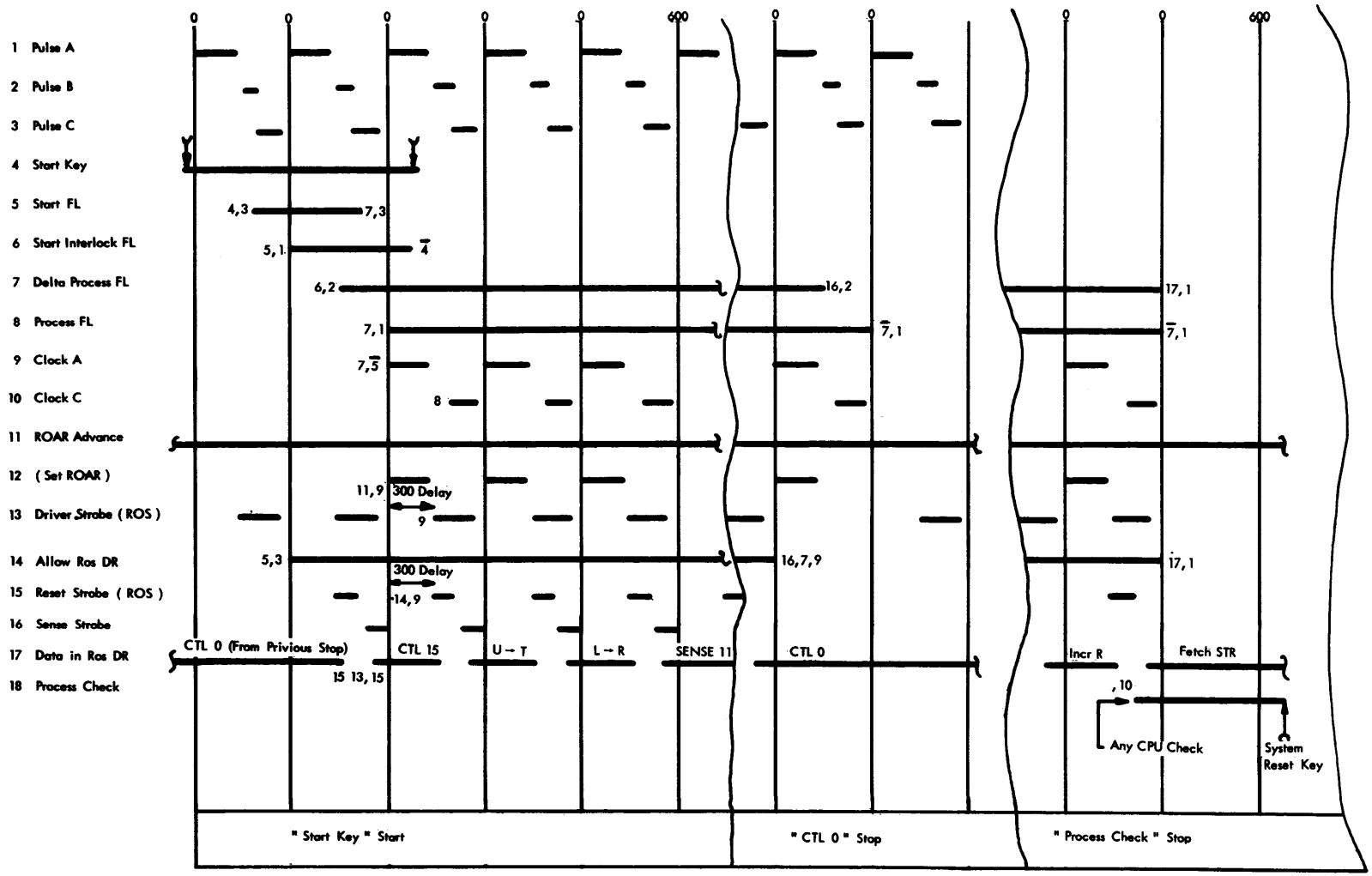


Figure 3-28 Start Key Timing

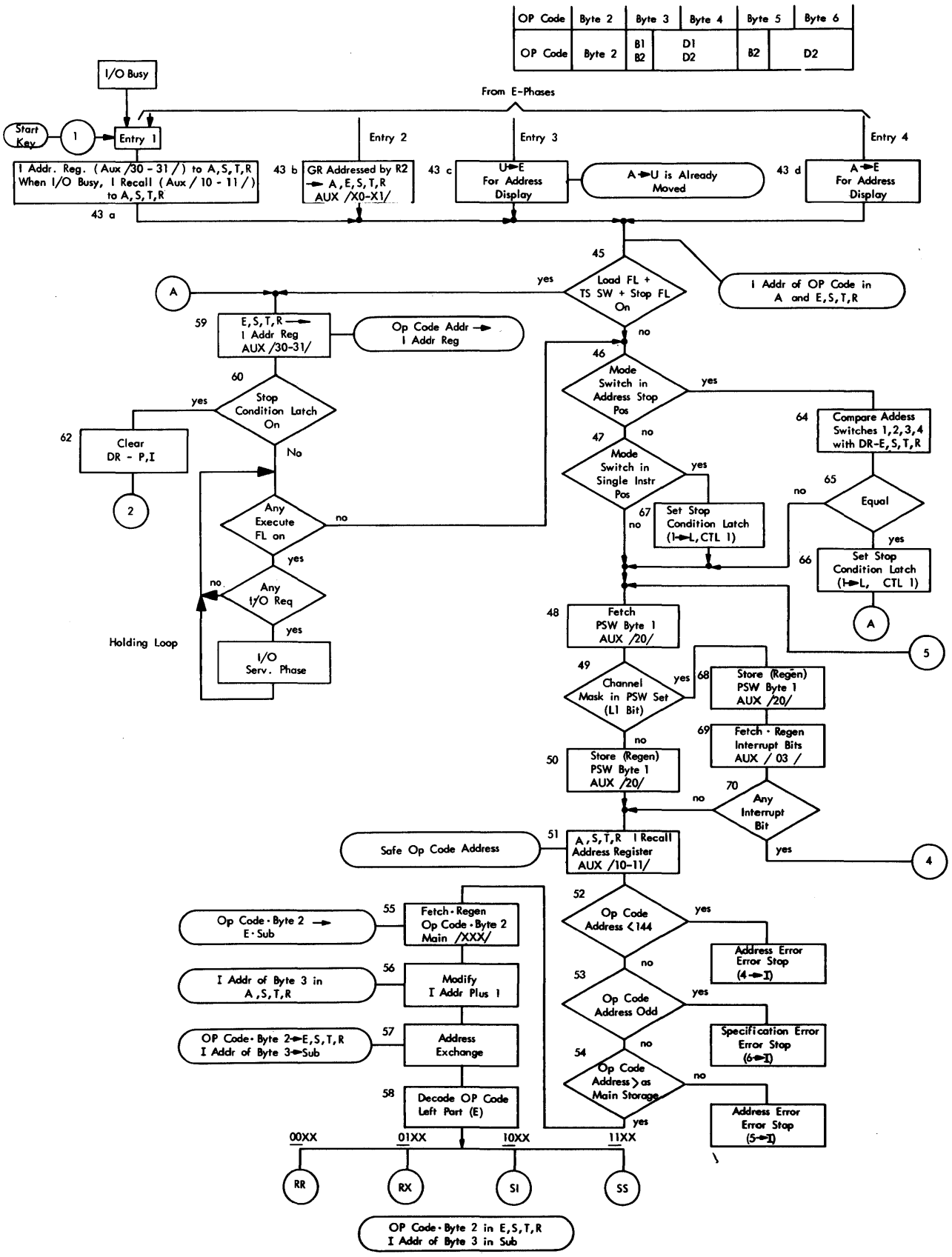


Figure 3-29 I-phase Entry

address of the NSI Op Code. This address is not in the I Address Register when Entry 2, 3, or 4 is used to enter the I-phase (Figure 3-29).

60. Test the on/off status of the Stop Condition FL. This is necessary because the Stop Key operation and the Address Stop operation share a common loop.
62. Clear DR-P and I for display purpose.
63. Read out Op Code addressed by DR-A, S, T, and R (Stop address) for display in DR-U and L.
10. The machine is stopped by a CTL 0 which resets the Process FL.

LOAD KEY (PROGRAM LOAD KEY)

- The purpose of the Load key is to read an 80 column card into main storage, thereby starting the machine program.
- The Load key starts the micro-program at its start address 3 F 3 F. Thus, the same sequence of operations occurs as in the System Reset Routine.
- During the Load Routine, the content of the (customer) Address Switches 1-4 (CPU-Console) is set into DR-U, S, T and R.
- The Load key initiates a Special E-phase in the addressed I/O device and thus a card is read in.

Description (Figure 3-27a)

The Load key reads an 80 column card into the main storage. This card contains instructions that enable the CPU to continue operations. Two prerequisites are necessary to successfully carry out the load operation:

1. The I/O Device (Card Reader) must be in the Ready state (Card in the hopper and Start key pressed).
2. The Mode switch must be in the correct position (Process, Address Stop or Instruction Step).

Up to step 11 in Figure 3-27, the Load routine is identical with the System Reset routine.

In step 11, the Load key is checked. If it was pressed, the Mode switch is checked to see whether it is in the correct position (-12-). It must be in either the Process, the Address Stop or the Instruction Step position. If it is in any of these positions, the content of Address Switches 1-4 (Customer Console) is set into DRU,S, T, R (-15-). These Address switches enable the customer to determine any desired start address for his program. The content of DR-S, T, R is set into the I-register. The I-register in Auxiliary Storage represents the second half of the PSW. Thus, the next sequential Op-Code address is stored (-16). As the DR-S, T, R are needed for further operations in the Load Routine, they are made available by an Address Exchange (-17-). The field length information is now set into E, S, T, R. This enables the card reader to read an 80 column card (-18-). A Sense 63 sets the necessary information into DR-L to enable the CPU to address the I/O device that is to read the card. When the I/O device is addressed, a further check is made to see whether the device is installed (-19-). If a non existent device were addressed, no operation occurs. The machine goes into the Read E-phase and the card is read into the main storage. Next the CPU goes into the I-phase to process the instruction contained in this card.

When the Transfer E phase for the CR (Console Reader) or MFCM I/O unit is executed, the micro-program enters the I phase Entry 1 as shown on Figure 3-29 to start the Load-program routine as given in the following steps.

43a. The start address is moved from the I Address Reg into DR-A , S, T, and R (by the subroutine "I Reg to E STR") for addressing the first Op Code in the Machine Load program. The contents of DR-A is set into DR-E for display purposes.
45. A Sense 15 tests for three conditions:
    1. Load Latch on?

2. Time Sharing switch off?
3. Stop Condition latch on?

If one of these conditions exists (in this example the Load latch is ON), continue as shown in step 59.

59. Store the Op Code address back into the I Address Reg with the Subroutine ESTR to Reg.
60. Test the on/off status of the Stop Condition latch, because the Stop Key and the Load Key share a common loop. Continue as shown in step 61.
61. The Holding Loop is a micro-program which prevents the CPU from continuing f rther into the I phase until the card which is presently being loaded is completely read in.

The Holding Loop is repeated as long as the Execute latch for any I/O unit is on. As soon as the Execute latch of the CR or MFCM is off (the first card is completely read), continue in the I phase as shown in step 46.

TIME SHARING SWITCH

- Objective: To enable CE to trouble shoot in I/O operations (Off position).

- If On, the execution of I/O operations (Service phases) is time-shared with CPU operations.

- If Off, the CPU processes an I/O operation completely (I phase, E phase, and all Service phases) before the next operation (I phase) is started.

Description

An I/O operation in the /360-20 is normally executed in the Time-Sharing-Mode (Time Sharing switch in the associated position). Time Sharing Mode enables the CPU to continue in the machine program as soon as the E phase, Not Service phases, of an I/O operation is executed. During the farther processing of the machine-program (I and E phases) the machine-program is interrupted each time a request from the running I/O unit is received. The request of an I/O unit requires the execution of an I/O service phase.

An I/O operation is very difficult to trouble shoot if the I/O operation is time shared with CPU operation.

The Time Sharing Switch in its off position controls the CPU to continue in the machine-program first when an I/O operation is completed with all Service phases.

To control the I/O operation (for example to check the contents of the read-in area in Main storage after a card is read in), the CPU can be stopped before the next I phase starts with the Mode switch in Address Stop position.

Micro-program Description (Figure 3-29)

45. In the next I phase, after the E phase of an I/O operation is executed, and the Time Sharing switch is in the off position, continue as shown in step 59 because:

Sense 15 tests for three conditions:
1. Load latch on?
2. Time Sharing switch off? (NO)
3. Stop Condition latch on?

59. Store the Op Code address of the instruction, following the I/O instruction, from DR-E, S, T, and R into the I Addr Reg in AUX-storage position /30-31/ with the subroutine ESTR Reg.
60. Test the on/off status of the Stop Condition latch. This is necessary because the Stop Key, Load Key, and the Time Sharing switch share a common loop. Continue as shown in step 61.
61. The Holding Loop is a micro program, which prevents the CPU from continuing farther into the I phase, until the started I/O unit is finished with all Service phases.

The Holding Loop is repeated as long as from any I/O unit the Execute latch is on. When the Execute latch of the I/O unit which was started by the previous instruction is off, continue in the I-phase as shown in step 46.

MODE SWITCH

Process

- Objective: To process machine instructions.

- The CPU starts processing when the Mode Switch is in Process position and the Start Key is pressed.

- The CPU starts processing with the I address which is available in the PSW (I Address Register).

- Processing is continued until any stop condition stops the CPU.

Description

Figure 3-30 is a flow chart showing the start of processing of a machine instruction. Note that the Mode

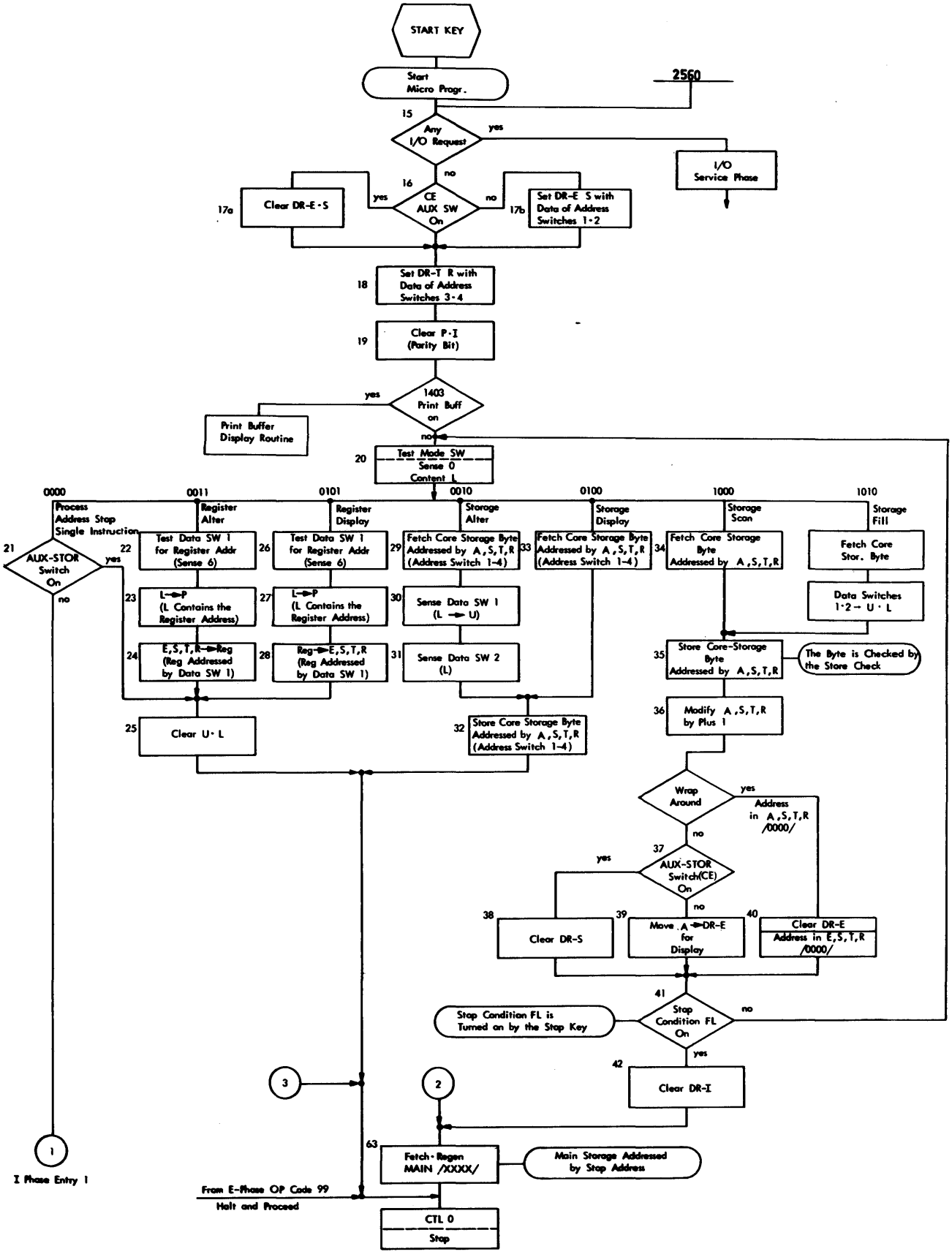


Figure 3-30 Manual Routine 1

switch must be in the Process position and the start key must be pressed before processing can begin.

15-19. Steps 15 through 19 are common for all Mode switch positions and perform no useful function for the Process operation.

20. Test Mode switch with a Sense 0. The bit pattern in DR-L is 1000 which directs the micro-program as shown in step 21.

21. The Test AUX-STOR switch must be off during the Process operation.

43a. The I-phase is entered by Entry 1 (Figure 3-29). The NSI Address located in the I-address Register is set into DR-A , S, T, R to address the NSI Op code.

Continue as shown in step 45 on Figure 145B. Description is in the section on CPU Operation on part I-phase.

Storage Display

- Objective: Move a byte from the core storage into DR-U, L for display.
- This operation is executed when the Mode switch is in the Storage Display position and the Start Key is pressed.
- The address of the byte is taken from the Address switches 1, 2, 3, and 4, and is set into DR- A , S, T, and R to address core storage.
- DR-P and I are cleared.
- When the AUX-STOR switch is off, a byte of Main storage is displayed. When the AUX-STOR switch is on, a byte of AUX storage is displayed.

Description (Figure 3-30)

Prerequisites:

1. Mode switch in Storage Display Position.
2. Address switches 1, 2, 3, and 4 set to the address of the byte to be displayed.
3. Start Key pressed.

16. Test condition of AUX-STOR Switch.

17a. AUX-STOR Switch on, set DR-E and S to zero because addresses equal or lower than /FF/ are used to address AUX storage. Any bit in E and/or S would address too big an address.

17b. AUX-STOR switch off, set the address part of Address Switches 1 and 2 into DR-E and S. DR-A is set when the address is higher than /FFF/.

18. Set the contents of Address Switches 3, and 4 into DR-T and R. DR-E, S, T, R now contain the address to be displayed.

19. Clear DR-P and I so that only a parity bit is displayed in these DR by Storage Display.

20. Test the Mode Switch. Continue as shown in step 33 if the Mode Switch is in the Storage Display position.

33. Read out Core Storage addressed by DR-A, S, T, R. The addressed byte is set into DR-U, L.

32. Store (Regenerate) the contents of DR-U, L into the same core storage position.

10. The CPU is stopped by a CTL 0 (Process FL reset).

17b. This detailed write up describes each micro-instruction required to set the address part of Address Switches 1 and 2 into DR-E and S, and to set the address part of Address Switch 1 into DR-A to address Main storage positions higher than /FFF/ (4K).

Micro-instruction	Remarks
Sense 2	Set the address part of Address Switch 1 into DL-L. Highest valid value with an 8K storage is 0000, 8K - 0001, 16K - 0011, 32 - 0111.
L - U	Save the address part of Address Switch 1 in DR-U.
Sense 3	Set the address part of Address Switch 2 into DR-L.
L - S	Move the address part of Address Switch 2 into DR-S. The address part of Address Switch 1 (in DR-U) is set into DR-A
U - E	Move the address part of Address Switch 1 into DR-E (for display only).

Storage Scan

- Objective: Check all bytes in Core-storage for correct (even) parity.
- The bytes are read out (one at a time in sequence) and stored back into Core-storage. If a byte does not have correct parity (even) the CPU stops with a Store Check.
- After the highest possible address of Core-storage is checked "Storage Wrap Around" occurs and the lowest /0000/ address is checked.
- When an error is present (Odd parity):
  1. DR-E, S, T, R displays the address of the incorrect byte.
  2. DR-U, L displays the incorrect byte.
  3. DR-P, I are cleared.

- The Storage Scan operation is executed when the Mode switch is in Storage Scan position and the Start Key is pressed.

- Main-storage is checked when the AUX-STOR switch is OFF.

- AUX-Storage is checked when the AUX-STOR switch is ON.

- If Core-storage contains no parity errors, the operation is stopped only by pressing the Stop Key.

Description (Figure 3-30)

Prerequisite: Mode switch in Storage Scan position and Start Key pressed (Figure 3-30).

15. A test for any I/O request is made because the CPU can be started either by pressing the Start Key or by any I/O request, that is, an I/O request is initiated by pressing the Printer Space Key or the MFCM NPRO Key. If any I/O request is present, the micro program continues in the corresponding (Printer or MFCM) service phase. If no I/O request is present, the micro program continues as shown in step 16.
16. Test condition of AUX-STOR switch.
17a. AUX-STOR switch ON (AUX storage is checked) clear DR-E, S because addresses equal or lower than /FF/ are used to address AUX-storage.
17b. AUX-STOR switch OFF, set the contents of Address switches 1 and 2 into DR-E, S. DR-A is set when the address is higher than /FFF/.
18. Set the contents of Address switches 3 and 4 into DR-T and R. DR-4K, S, T, and R now contain an address to address Core storage.
19. Clear DR-P and I so that only a parity bit is displayed when the CPU stops.
20. Test Mode switch. Continue as shown in step 34 if the Mode switch is in the Storage Scan position.
34. Read out into DR-U and L, the Core-storage position addressed by the contents of DR-A, S, T, R.
35. Store the contents of DR-U and L back into the addressed Core storage position. If the stored byte does not have even parity (DR-U and L should both be odd so the whole byte is even), the Store Check FL is turned on and the CPU stops. The contents of this byte can now be altered by a Storage Alter operation.
36. Modify the address in DR-A, S, T, R plus 1 (Subroutine STR + 1) to address the next byte.

    When the highest possible address (for an 8K storage/1FFF/) is present and the modification plus 1 is executed, the address is modified to /0000/.

    The modification from the highest to the lowest address is called Wrap Around. In all operations other than Storage Scan, a Wrap Around condition stops the CPU and an address error is indicated (DR-I contains /5/).

    If a Wrap Around occurs during the Storage Scan operation, continue as given in step 40, if no Wrap Around occurs continue as given in step 37.
40. When the Wrap Around condition occurs DR-E is set to /0/ DR-E, S, T, R now contains /0000/ for address display if an error occurs. Continue as given in step 41.
37. When no Wrap Around condition occurs, the AUX-STOR switch is tested. If off, continue as given in step 39; if on, continue as given in step 38.
38. To check the bytes in AUX-storage with the Storage Scan routine, the AUX-STOR switch must be set to the ON position.

    The AUX-storage contains 256 bytes which are addressed by addresses from /00 to FF/. To address an AUX-storage position, the address is located in DR-S, T and R (DR-S must be /0/). If the highest address for AUX-storage /FF/ is present in DR-T and R and this address is modified by the Subroutine STR + 1 as shown in step 36, DR-S, T, and R contains /100/. To address the lowest AUX storage position (/00/) after addressing, the highest AUX-storage position (/FF/) DR-S which contains /1/ must be cleared.

    Actually DR-S is cleared after each modification, and the addresses in T and R are modified from /00/ step by step to /FF/ and from /FF/ direct to /00/. The modification of AUX storage address from /FF/ to /00/ can be called AUX wrap around.
39. When the address in DR-A, S, T, R which is addressing Main-storage is modified plus 1 as described in step 36 (Wrap Around), DR-E is not updated with the modified address. The contents of DR-A are moved into DR-E for address display if an error occurs. Continue with step 41.

41. Test Stop Condition FL.

If the Stop Key has not been pressed to end the Storage Scan operation, the next byte of Core-storage is checked. Continue with step 34.

If the Stop Key has been pressed to end the Storage Scan operation, the Stop Condition FL is on.

42. DR-I is cleared to parity bit only. DR-I was reset as described in step 19, but was used in the S, T, R + 1 Sub routine.

63. Read out and regenerate the byte in whatever position the machine stops to display this byte in DR-U, L.

10. The CPU is stopped by a CTL 0 (Reset Process FL).
    a. DR-E,S,T,R contains the stop address.
    b. DR-U, L contains the byte of the stop address.
    c. DR-P, I are cleared.

Address Stop

- Objective: Process machine instructions until the Op code address in the machine program is equal to the address in Address switches 1, 2, 3, and 4.

Description

Prerequisites:

1. Mode switch in Address Stop position.
2. Address switches 1, 2, 3, and 4 must be set equal to an address which contains an Op code of the machine program.
3. Machine program must be running. At the beginning of each I-phase the Address Stop switch is tested. For the following steps refer to Figure 3-29.

46. A test is made to see if the Address Stop switch is on; if it is, continue as shown in step 64.

64. DR-E, S, T and R contain the address of the Op code of the instruction which is decoded during this I-phase. The address in E, S, T, and R is compared with the address set in the Address switches 1, 2, 3, and 4.

If the address in the Address switches is not equal to the address in E, S, T and R, the CPU does not stop but continues with the I-phase as shown in step 48.

If the addresses in the Address switches and in E, S, T, R are equal, the Stop condition FL is turned on as shown in step 66.

66. The Stop Condition FL is set by two micro-instructions.
    a. a /1/ is moved into DR-L.
    b. a CTL 1 turns on the Stop Condition FL when DR-L contains a /1/.

59. Sub routine E, S, T, R Reg moves the stop address in E, S, T, R into the I Address Register in PSW byte 3, 4 (AUX position /30-31/.

It is necessary to save the stop address in the I Address Reg., because when the CPU stops, the address in E, S, T, R can be destroyed by a Storage Display routine. The CPU must be able to continue in the machine program when the Start Key is pressed after the machine stops due to the Address Stop routine.

60. Test Stop Condition FL; if On, continue as shown in step 62.

62. Clear DR-P, I.

63. Fetch (and regenerate) Op Code addressed by the stop address into DR-U and L to display the bit configuration of the Op Code.

10. The CPU is stopped by a CTL 0 resets (turns off) the Process FL.
    a. DR-E, S, T, and R contain the address at which the CPU stopped.
    b. DR-U and L contain the Op code of the stopped address.
    c. DR-P and I are cleared.

Register Alter

- Objective: Replace the contents of one GR or Address register located in AUX-storage location /X0-X1/ with the data which is set in Address switches 1, 2, 3, and 4.

- This operation is executed when the mode switch is in Register Alter position and the Start Key is pressed.

- The address of the register which is to be altered is taken from Data switch 1.

- DR-P contains the address of the altered register.

- DR-U and L are cleared.

Data switch positions (addresses) and the associated registers are:

Position of Data Switch 1	Addressed Register
0	Op. Code · Byte 2 Register
1	I Recall Address Register
2	PSW Byte 1 · 2
3	PSW Byte 3 · 4 (I Address Reg.)
4	Operand 1 Address Reg.
5	Operand 2 Address Reg.
6	Operand 2 Recall Reg.
7	Print Interrupt Address
8	GR 8
9	GR 9
A	GR 10
B	GR 11
C	GR 12
D	GR 13
E	GR 14
F	GR 15

Description (Figure 3-30)

Prerequisites:

1. Mode switch in Register Alter position.
2. Address switches 1, 2, 3, and 4 set to the data which it is desired to set into the addressed register.
3. Data switch 1 is set to the address of the register that it is desired to Alter.
4. Start Key pressed (Figure3-30).

15. First, the micro-program tests for any I/O request because the CPU can be started either by pressing the Start Key or by any I/O request (that is, an I/O request is initiated by pressing the Printer Space Key or the MFCM NPRO Key). If any I/O request is present, the micro-program continues in the corresponding (Printer or MFCM) service phase. If no I/O request is present, the micro-program continues as shown in step 16.
16. The condition of the Auxiliary Storage Switch is tested because it must be off in order to accomplish a Register Alter operation.

17b. If the Auxiliary Storage switch is off, the data from Address Switches 1 and 2 are set into DR-E and S.

18. The data from Address Switches 3 and 4 are set into DR-T and R. With the last two operations, the data from Address Switches 1, 2, 3, 4 were set into the DR-E, S, T, R.
19. Zeros are set into the DR-P and I, so that only the parity bit is turned on.
20. Next, a test is performed by the micro-program, to determine whether a print buffer scan is on.
21. If a print buffer scan is on, the micro-program goes into the print buffer routine.
22. If no Print Buffer Scan is on, the micro-program tests the Mode switch. This is necessary because up to this point the micro-program routine is common to all the different display and alter operations (Common loop of the Start key).
23. If the Mode switch is in the Register Alter position, the micro-program starts the actual Register Alter Routine by testing Address Switch 1. A Sense 6 operation moves the Address which is set up in Address Switch 1, into DR-L. DR-L now contains the address of the General Register which is to be altered.
24. The content of DR-L is then moved into DR-P, so the Auxiliary Storage can be addressed with an SPN operation.
25. The E, S, T, R to Register micro-program sub-routine moves the data from DR-E, S, T, R into the Register which is addressed by DR-P.
26. Since the micro-program Sub-routine made use of the DR-U and L in accomplishing the ESTR to Register operation, DR-U and L are set to zero. This is necessary to avoid confusion when looking at the display panel.

10. A Control 0 operation stops the CPU. This ends the Register Alter operation. DR-P contains the address of the register just altered and DR-E, S, T, R contain the data, which are now stored in the altered register.

Storage Alter

- Objective: Set the data in Data Switches 1 and 2 into one byte of Main Storage.

- This operation is executed when the Mode switch is in the Storage Alter position and the Star Key is pressed.

- The address of the byte is taken from the Address switches 1, 2, 3, and 4 and is set into DR-A, S, T, and R to address Core-storage.

- DR-P and I are cleared.

- When AUX-STOR switch is off, the data in Data switch 1 and 2 is set into main storage. When AUX-STOR switch is on, the data is set into AUX-storage.

Description (Figure 3-30)

Prerequisites:

1. Mode Switch in Storage Alter position.
2. Data switch 1 and 2 set to the data which is to be set into Core-storage.
3. Address switches 1, 2, 3, and 4 set to the address that it is desired to alter.
4. Start Key pressed (Figure 3-30).

15. Test for any I/O request because the CPU can be started either by pressing the Start Key or by any I/O request (that is, Printer Space Key, or MFCM Non Process Run Out Key). If any I/O request is present, the micro-program continues in the corresponding (Printer or MFCM) service phase. If no I/O request is present, the micro-program continues as shown in step 16.
16. The Auxiliary Storage switch determines whether a byte of the Auxiliary Storage (Switch on) or a byte of the main storage is going to be altered. When the Start key is pressed, the Storage Alter operation is initiated (Figure 3-30. The micro-program first performs the common start-routine. When the Auxiliary Store switch is tested there are two different operations possible, 17a or 17b:

17a. If the Switch is on, DR-E and S are set to zero because only addresses lower or equal to /FF/ are used to address Auxiliary Storage.

17b. If the Switch is off, the content of Address Switches 1 and 2 is set into the DR-E and S.

18. The content of Address Switches 3 and 4 is set into DR-T and R. The address of the byte, which is to be altered, is now either in DR-T and R or in DR-E, S, T, R.
21. The test of the Mode switch directs the micro-program to the actual Storage Alter Routine.
29. The micro-program reads out the byte which is addressed by DR-S, T, R (or if the address is higher than /FFF/ by DR-A, S, T, R) into DR-U and L. Thus, the byte is completely cleared.
30. The micro-program moves the data from Data Switch 1 into DR-L. The content of DR-L must next be moved into DR-U because DR-L is still needed for the following Sense operation.
31. The next Sense instruction sets the data from Data Switch 2 into DR-L. DR-U and L now contain the data from Switches 1 and 2.
32. With a "Store STR" operation the micro-program sets the content of DR-U and L back into the previously cleared byte. Thus, the byte is altered.
10. A Control 0 operation stops the CPU. DR-E, S, T, R contain the address of the altered byte. DR-U and L contain the bit configuration of the altered byte. DR-P and I are set to zero to avoid confusion when looking at the display panel.

Register Display

- Objective: Move the contents of one GR or Address register (16) located in AUX-storage location /X0-X1/ into DR-E, S, T, and R for display.
- This operation is executed when the Mode switch is in the Register Display position and the Start Key is pressed.
- The address of the register which is to be displayed is taken from Data switch 1.
- DR-P contains the address of the displayed register.
- DR-U and L are cleared.

Position of Data Switch 1	Displayed Register
0	Op Code · Byte 2 Register
1	I Recall Register
2	PSW Byte 1 · 2
3	PSW Byte 3 · 4 (I Address Reg.)
4	Operand 1 Address Reg.
5	Operand 2 Address Reg.
6	Operand 2 Recall Reg.
7	Print Interrupt Address
8	GR 8
9	GR 9
A	GR 10
B	GR 11
C	GR 12
D	GR 13
E	GR 14
F	GR 15

Description (Figure 3-30)

Prerequisites:

1. Mode switch in Register Display position.
2. Data switch 1 is set to the address of the register which is to be displayed.
3. Start Key pressed (Figure 3-30).

15. Test for any I/O request because the CPU can be started either by pressing the Start Key or by any I/O request (that is, an I/O

request is initiated by pressing the Printer Space Key or the MFCM NPRO Key). If any I/O request is present, the micro-program continue in the corresponding (Printer or MFCM) service phase. If no I/O request is present, the micro-program continue as shown in step 16.

16. Step 16-18 are idle steps for the Register Display routine (common Start Key routine for all Mode switch positions).
19. Clear DR-P and I so that only a parity bit is displayed.
20. Test the Mode switch. Continue as shown in step 26 if the Mode switch is in the Register Display position.
26. Test Data switch 1 for the address of the register which is to be displayed. A Sense 6 moves the data (address) in Data switch 1
27. Move the contents of DR-L (address of register which is displayed) into DR-P so that the AUX storage can be addressed by a FPN micro-instruction.
28. Move the contents of the addressed register into DR-E, S, T, and R by using the Sub-routine Reg - E, S, T, R.
25. Clear DR-U and L so that only a parity bit is displayed.
10. The CPU is stopped by a CTL 0 which resets the Process FL.
    a. DR-P contains the address of the displayed register.
    b. DR-E, S, T, and R contain the contents of the displayed register.
    c. DR-U, L, and I are cleared.

Storage Fill

- Objective: The Storage Fill operation is a CE-aid which enables the operator to fill all bytes of the core storage with a bit pattern. Storage Fill provides a quick method of checking the core storage.

- The operation is executed when the Start key is pressed with the Mode switch in the Storage Fill position.

- According to the position of the Auxiliary Storage switch, either the Auxiliary or the Main Storage is filled with the desired bit pattern.

- The start address (the address of the byte, which is to be filled first) is set with Address switches 1-4..

- The fill-bit pattern is set with the Data switches 1 and 2.

- The fill operation is executed continuously until the Stop key is pressed.

Description (Figure 3-30)

When the start key is pressed, the micro-program first executes all the operations in the common start loop.

15. The micro-program checks for Any I/O Request. If an I/O Request is present, the micro-program continues in the corresponding Printer a MFCM service phase.
16. If there is no I/O Request, the micro-program tests the CE Aux Store switch.
17. If the CE Aux Store switch is On, DR-E and S are set to zero because only addresses lower or equal to /FF/ are used to address Auxiliary Storage.
17b. If the CE Aux Store switch is off, the data in Address switches 1 and 2 are set into DR-E and S.
18. The micro-program sets the data from Address switches 3 and 4 into DR-T and R. Thus, the Start Address is either in DR-T and R or in DR-E, S, T, R.
19. Next, DR-P and I are set to zero so they contain Parity Bits only.
20. A test is performed to see whether a print buffer scan is up. If it is up, the micro-program continues in the print buffer CE routine.
21. If there is no print buffer scan, the Mode switch is tested to direct the micro-program to the desired routine.
70. When the Mode switch is in the Storage Fill position, the micro-program fetches a byte according to the address in S, T, R. This byte is set into DR-U and L.
71. With 2 Sense operations the micro-program moves the data from Data Switches 1 and 2 into DR-U and L. DR-U and L now contain the desired bit configuration.
35. The content of DR-U and L is stored back into the core storage position which is addressed by DR-S, T, R (or A, S, T, R).
36. The content of DR-S, T, R is next modified by one to enable the micro-program to fetch the next byte (In this manner all bytes are filled step by step). After each modification (Subroutine STR + 1), the micro-program checks for Storage Wrap Around.

40. If there is a Storage Wrap Around, the content of the DR-A has no display facilities. Therefore, any error would be displayed in DR-E.
37. If there is no Storage Wrap Around the micro-program first tests the Auxiliary Storage switch (CE) before it sets the content of DR-A into DR-E.
39. If the Auxiliary Storage switch is off, the micro-program sets the content of DR-A into DR-E for display purposes.
38. If the Aux Switch is On, DR-S is set to zero to obtain the lowest Auxiliary Storage Address.
41. The Stop condition latch is tested to see whether the Stop key was pressed. If it was not pressed, the micro-program goes back to step 21 to test the Mode switch. If the Mode switch is in the Storage Fill position, the same sequence of operation continues until the Stop key is pressed.
42. If the Stop key was pressed, DR-I is set to zero since DR-I was used by the micro-program for the STR+1 subroutine.
63. The micro-program fetches and restores one byte, which is addressed with the current address in S, T, R. In this manner, the Stop Address is displayed in DR-U and L.
10. A Control 0 operation stops the CPU and ends the Storage Fill. When the Storage Fill operation is terminated, the DR-S, T, R contain the Stop Address. DR-U and L contain the bit configuration of the byte which was addressed by the Stop Address. DR-P and I display only the parity bit (To avoid confusion when looking at the display panel).

Single Instruction

- **Objective: Process one complete machine instruction (I and E phase) and to stop the CPU before the next I phase is executed.**

Description (Figure 3-30)

Prerequisites: Mode switch in Single Instruction position, Start key pressed (Figure 3-30). Steps 15 through 19 perform no useful function for the Single Instruction operation.

20. A test of the Mode switch is performed to direct the micro-program to the Single Instruction routine.
21. The Auxiliary Storage switch is tested next to insure that it is turned off. It must be off to enable the micro-program to address Main Storage during the Single Instruction Routine.

43a. The I-phase is entered at Entry 1. The address of the NSI, which is located in the I-address register (Position /30-31/ of Aux) is moved into DR-A, E, S, T, R to enable the micro-program to read out this instruction, when it is needed.
45. The micro-program checks three conditions: Load Latch On, Not Time Sharing Switch and Stop Latch On.
59. If any one of these conditions is present, the NSI address is stored back into the I-address register and a further test is performed to see whether the Stop Condition latch is on.
62. If the Stop Condition latch is On, DR-P and I are set to zero and the CPU stops. Thus, the CPU always stops at the beginning of an I-phase when the Stop condition latch was turned on previously.
60. If the Stop Condition latch is Off, either the Time Sharing switch is Off or the Load latch is On. In either case, the micro-program starts a special Sub-routine, the Holding Loop. The micro-program stays in the Holding Loop until either the Time Sharing switch is turned On and/or the Load latch is turned Off.
46. If the Load latch is not On, the Time Sharing switch is in the TS position and the Stop latch is not On, the Mode switch is tested to insure that it is not in the address Stop position. This test is necessary because the micro-program routine described here is common for several other operations.
47. For the same reasons, the Mode switch is tested again. It must be in the Single Instruction position.
67. With the Mode switch in Single Instruction position, the Stop Condition latch is set. This enables the CPU to stop at the beginning of the next I-phase. In this manner, only a single instruction (I-phase and E-phase) is processed.
48. The micro-program continues with fetching the first byte of the PSW.
49. A test is performed to see whether there is a channel mask set (A one bit in DR-L). If there is a channel mask set, the micro-program is directed to the interrupt routine.
50. When there is no channel mask present, byte one of the PSW is stored back into Auxiliary Storage.
51. The address of the NSI, which is in DR-A,

E, S, T, R, is set into the I-Recall Address Register (Position /30-31/ of Aux.) to save it so that an operation can be repeated when a micro-program error stop occurs during either the I- or the E-phase of an operation.

52. The NSI Address addresses one byte in the Main Storage. This byte is the first (leftmost) byte of the instruction. The first byte of an Instruction is its Op-code. The NSI Address is actually the address of the Op-code of the Next Sequential Instruction. This Op-code Address is tested to insure that it is correct. The address must be equal to or greater than /090/. If it is less than /090/, the protected area of the main storage is illegally addressed and the CPU stops on an address error. In that case, a 4 is set into DR-I to display the error number. The error number is used to identify the error by means of an error list.
53. The Op-code Address is further tested to make certain that it is even. The units position of the address must be /0, 2, 4, 6, 8, A, C, E/. If the address is not even, the CPU stops on a Specification Error. A 6 is moved into DR-I.
54. Finally, the Op-code Address is tested to see whether it is outside available storage. It must not exceed /FFF/ for a 4K storage, /1FFF/ for an 8K storage, /2FFF/ for 16K storage or /3FFF/ for a 32K storage. When the address is not within these limits, the CPU stops on an Address Error and a 5 is set into DR-I.
55. When there was no error, the micro-program reads part of the Next Sequential Instruction out of Main Storage. To accomplish this, the micro-program uses the Memory to E, S, T, R Sub-routine. In this Sub-routine, a series of micro-instructions reads out the Op-code, performs an address exchange and increases DR-R by 1 to fetch byte two of the NSI and so on. When the Memory to E, S, T, R operation ends, the Op-code and byte 2 are in DR-E, S, T, R and byte 3 of the NSI is in the associated Sub-registers.
56. The OP-code is again tested for validity and subsequently its format is decoded. Each Op-code has its own E-phase, which now follows the common I-phase. After the E-phase, the I-phase is reentered and the CPU stops because the Stop Condition latch was previously turned on.

CPU OPERATION

- A machine program is carried out by consecutive machine instructions.
- Each machine instruction is divided into two phases.
  1. The Instruction (I) phase
  2. The Execute (E) phase
- The I phase micro-program routine is common for all machine instructions.
- Each machine instruction has its own micro-program routine for the E phase.

I-PHASE OPERATIONS

In the I phase micro-program routine, the Op Code register and Address registers in the AUX storage are loaded with the Op code and the addresses of the machine instruction which is to be executed after the I phase. In preparation for starting an I phase, the address of the Op code (NSI address) is placed into DR-A, S, T, and R to read out the Op code. The contents of DR-A is placed in DR-E for display purposes.

The I phase micro-program routine can be entered via four paths depending upon the previous operation (E phase).

The four entries (labeled Entry 1 to 4 on Figure 3-29) are required because, at the end of an E phase, the NSI can be located in different places.

I phase entry Flow Chart (figure 3-29). Entry 1 is used, when the NSI address is located in the I Address Reg (AUX storage position /30-31/).

The I phase is entered by Entry 1 if the Start Key is pressed to start the machine program or when an I/O unit is selected from the machine program but is still busy or when the previous E phase has processed one of the following Op Codes /1A/, /1B/, /40/, /48/, /49/, /4A/, /4B/, /91/, /92/, /94/, /95/, /96/, /D0/, /D1/, /D2/, /D3/, /D5/, /DC/, /DE/, /FC/, /FD/, /F1/, /F2/, /F3/, /F8/, /F9/, /FA/, or /FB/.

43a. The NSI Address is moved from the I Address Reg. into DR-A, S, T, and R (by the

Subroutine "I Reg. to E STR") for addressing the Op -code in main storage. The contents of the DR-A are set into DR-E for display purposes.

NOTE: If Entry 1 is entered, via the Busy I/O routine, the I Recall Addr Reg and not the I Addr Reg is moved into DR-A, S, T, and R in order to try to accomplish the I/O operation again. For further detailed description see "Transfer E phase" in the various I/O units.

Entry 2 is used when the NSI address is located in a GR. The NSI address is located in a GR when the previous E phase processed an Op code /07/ or /0D/ and a Branch occurred.

43b. The NSI address stored in a GR is moved by the Subroutine I Reg. to E-STR into DR-A, S, T, and R for addressing the Op Code in Main storage. The contents of the DR-A are set into DR-E for display purposes.

Entry 3 is used, when the NSI address is located in DR- A, S, T, and R and the contents of the DR-A are stored in DR-U. This condition exists when the previous E phase processed an Op code /07/ no branch, /0D/ no branch, /46/ branch, /47/ no branch, /4D/. DR-U is moved into DR-E for display purposes.

43c. The contents of DR-U (DRA) are moved into DR-E for display.

Entry 4 is used, when the NSI address is located in DR- A, S, T, and R (the contents of DR-A are not stored in DR-U. See Entry 3). This condition exists when the previous E-phase has processed an Op code /47/ branch, /81/. The contents of DR-A are transferred into DR-E for display purposes.

43d. The contents of DR-A are moved into DR-E for display purposes.

45. A Sense 15 tests for three conditions:
   a. Load latch on.
   b. Time Sharing switch off.
   c. Stop condition latch on.

   If none of these conditions exist, continue with step 46.

   If one or more of these conditions exists continue as shown in step 59.

46. Test for the on/off status of the Address Stop switch and if it is not on, continue as shown in step 47. Address Stop functions are described in the section on Address Stop.

47. Test for the on/off status of the Single Instruction switch and if it is on (normal case) continue as shown in step 48. Single Instruction mode functions are described in the section on Single Instruction.

48. Read out PSW byte 1 AUX pos. /20/ to test for the presence of a channel mask. When the channel mask is present, DR-L contains a 1 bit.

49. Test for the presence of a channel mask and if there is none present, continue as shown in step 50, if one is present, see the section on Interrupt routine.

50. Store PSW byte 1. PSW byte 1 is regenerated because it is used again in the next I phase.

51. Move the contents of DR-A, S, T, and R (Op Code address) into the instruction (I) Recall Reg AUX position /10-11/ with the Subroutine S, T, R to I Recall Reg.

   The I Recall Register saves the Op code address so that an operation can be repeated (I and E phase) when a micro-program error stop occurs during either the I or the E phase of an operation.

52. Test the contents of DR-A, S, T, and R (Op Code address) for an address equal to, or greater than, /090/. If the address is equal to, or greater than /090/, continue as shown in step 53. If the address is less than /090/, the protected area of the main storage is illegally addressed, the CPU stops, and an address error is indicated by DR-I containing a /4/.

53. The contents of DR- A, S, T, and R (Op Code address) are tested to make certain it is an even address (units position of the address must be /0, 2, 4, 6, 8, A, C, or E/. If the Op Code address is even, continue as shown in step 54. If it is odd, the CPU stops with a Specification Error (DR-I contains /6/).

54. The contents of DR- A, S, T, R (Op Code address) are tested to see if they contain an address outside available main storage. For a 4K CPU the address should not be greater than /FFF/; 8K - 1FFF; 16K - 2FFF; 32K - 3FFF.

   If the address is within the CPU capacity, continue as shown in step 55. If the address is greater than the CPU capacity the CPU stops with an Address error (DR-I contains /5/).

55. The addressed Op code and Byte 2 of the machine instruction are read out of Main

Storage (with the Subroutine Mem - E STR). The content of DR-R is modified plus 1 to update the address to the address of Byte 3, in the instruction.

57. An address exchange (CTL 15) is executed to set the lower part of the Op code and Byte 2 from the Subregister into DR-S, T, and R for testing purpose.
58. The format of the Op code is decoded by testing DR-E (upper part of the Op code) to determine which of the four formats RR, RX, SI, or SS is associated with the machine instruction.

    Following this decoding DR-E, S, T, and R contain the Op code and Byte 2 of the machine instruction and the Subregister contains the I address of Byte 3.

RR Format Op Code Decoding

- There are 7 valid Op codes in the RR Format.
- The RR Format is already decoded in the previous part of the I-phase (Figure 3-29).

Description (Figure 3-31)

1. Test for Branch Op codes.

   The upper part of the Op code is in DR-E. If DR-E contains a 1 bit (Not a Branch Op Code), continue as shown in step 5. If DR-E contains no 1 bit (Branch Op Code), continue as shown in step 2.
2. Test Branch Op code for a Valid R2 address. The R2 field contains the address of a GR, but, because a GR can only be addressed by an address higher than /7/, the R2 address is tested for validity (/8/-/15/). When the R2 address is valid, DR-R contains an 8 bit.

   If DR-R contains an 8 bit, continue as shown in step 4.

   If DR-R contains no 8 bit, continue as shown in step 3.
3. In a Branch Op code, the R2 address can be /0/ to represent an unconditional branch.

   If the R2 address is not /0/ (and it is also not greater than /7/), the CPU stops. DR-I contains a /6/ to indicate a Specification error (See Figure 3-27). If the R2 address is /0/, continue as shown in step 4.
4. DR-S is tested to see if it contains an 8 bit. Continue as shown in step 6 if DR-S contains an 8 bit. An 8 bit in DR-S indicates that the 8 bit Op code is /0D/ (Branch and Store) and for this Op Code the R1 address is to be tested for validity (step 6).

   R1 is an address of a GR and must be (1XXX). If DR-S does not contain an 8 bit, continue as shown in step 7.
5. Test the Op code (non branching) for a valid R2 address. The R2 field contains the address of a GR, but because a GR can only be addressed by an address higher than /7/, the R2 address is tested for validity. If the R2 address is valid DR-R contains an 8 bit.

   If DR-R contains an 8 bit, continue as shown in step 6.

   If DR-R contains No 8 bit, the CPU stops. DR-I contains a /5/ to indicate an Address error (See Figure 3-27a).
6. The R1 field contains the address of a GR, but because a GR can only be addressed by an address higher than /7/, the R1 address is tested for validity (/8-15/). If the R1 address is valid, DR-T contains an 8 bit.

   If DR-T contains an 8 bit, continue as shown in step 7.

   If DR-T contains no 8 bit, the CPU stops. DR-I contains a /5/ to indicate an Address error (See Figure 3-27a).
7. The Op code, located in DR-E and S, is stored in the Op Register in AUX-storage position /00/ for later use.

   NOTE: Steps 08 through 11 are common to all Formats (RR, RX, SI and SS) and are therefore not accomplished in the most logical way for the RR format.

8. An address exchange moves the NSI address from the Subregister into DR-A,S, T, and R
9. The contents of DR-A,S, T, and R (NSI address) is stored (Subroutine S, T, R Reg) in the I Address Reg AUX storage position /30-31/ to save it for the next I-phase.
10. An address exchange moves the NSI address into the Subregister for later use.
11. The Op code is read out of the Op Reg. in AUX storage position /00/ into DR-U and L. The contents of DR-L (lower part of Op Code) is moved into DR-E for later use.
12. The contents of DR-U (upper part of Op Code) is tested by a Use DR-U for further decoding.

    If the contents of DR-U is either 0011 or 0010, the Op code is invalid and the CPU stops. DR-I contains a /1/ to indicate an Invalid Op code (Figure 3-27a).

    If the contents of DR-U is 0001, continue as shown in step 13.

    If the contents of DR-U is 0000, continue as shown in step 17

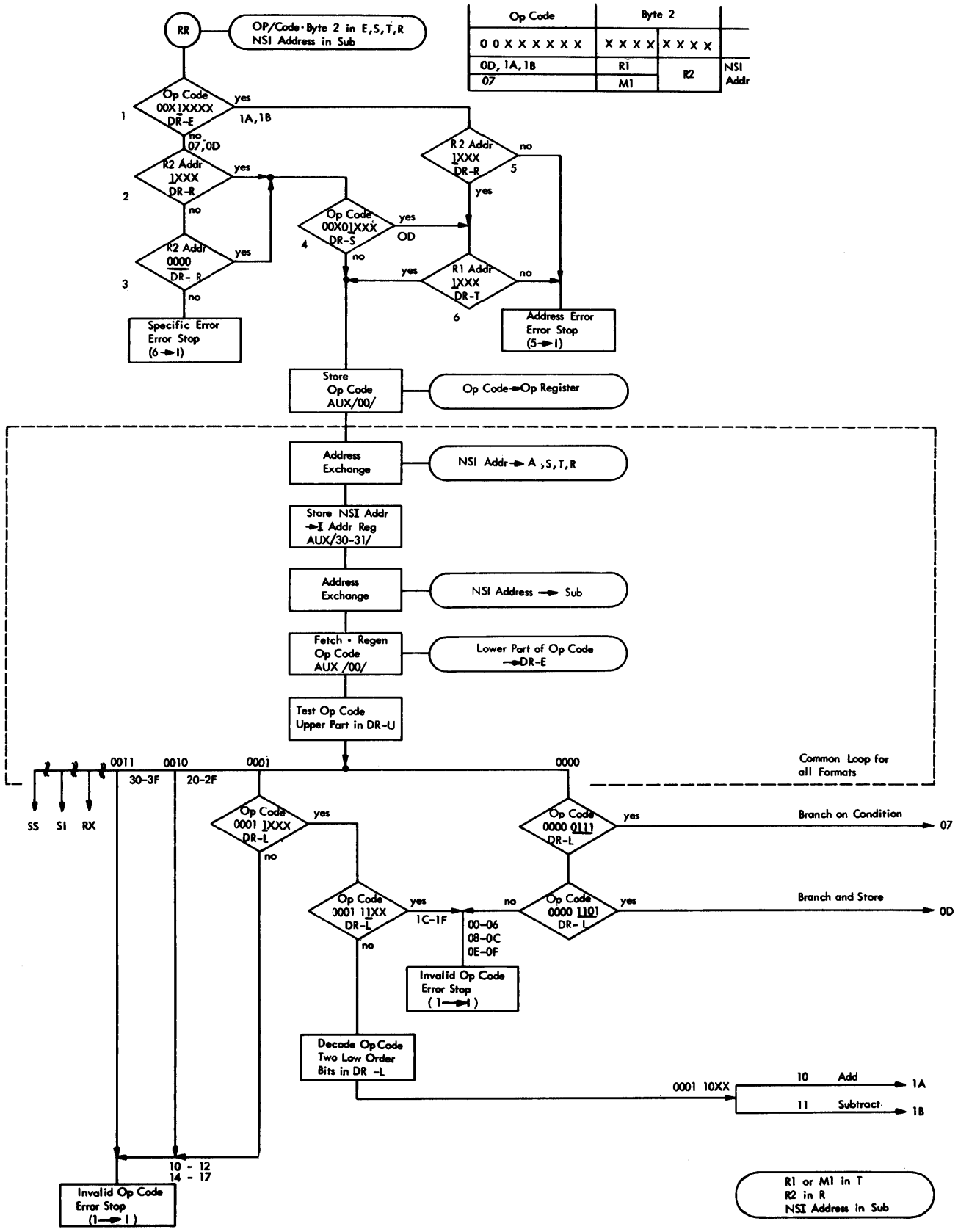


Figure 3-31 I-phase Op Decoding RR

13. Test DR-L for the presence of an 8 bit. If an 8 bit is present, proceed to step 15. If no 8 bit is present, proceed to step 14.
14. Test the bit pattern in DR-L.

    If the contents is not 0011, the Op code decoded is invalid and the CPU stops. DR-I contains a /1/ to indicate an Invalid Op code (Figure 3-27a).
15. Test DR-L for the presence of a 4 bit. If DR-L contains a 4 bit, the Op code is invalid and the CPU stops. DR-I contains a /1/ to indicate an Invalid Op code error stop (Figure 3-27a).

    If DR-L does not contain a 4 bit, bit 2 and bit 1 of DR-L are decoded and the Op Code is determined as shown in step 16.
16.

    If DR-L contains no 1 bit and a 2 bit = Op Code Add /1A/.

    If DR-L contains a 1 bit and a 2 bit = Op Code Subtract /1B/.

    When the Op code is determined (decoded), the associated E-phase is executed.
17. Test the contents of DR-L for 0111, which, if present, signifies that a Branch on Condition /07/ Op Code is decoded and the E-phase of this Op code starts.

    If the contents of DR-L is not 0111 proceed to step 18.
18. Test the contents of DR-L for 1101, which, if present signifies that a Branch and Store /0D/ Op Code is decoded and the E-phase of this Op code starts.

    If the contents of DR-L is not 1101, it contains an invalid Op Code and the CPU stops. DR-I contains a /1/ to indicate an Invalid Op Code error stop (Figure 3-27a).

RX Format Op Code Decoding

- There are 12 valid Op Codes in the RX Format.
- The RX Format is determined in the first part of the I-phase (Figure 3-29).

Description (Figure 3-32)

1. Test for a valid R2 field in DR-T.

   For all Op codes in the RX Format, the R2 field is not used and should contain a /0/.

   If the R2 field is not /0/, the CPU stops and DR-I contains a /6/ to indicate a Specification error (Figure 3-27a).
2. Test for Op code Branch on Condition /47/.

   Op code /47/ has an M1 field (Mask) which can contain any combination between /0/ and /15/.
3. Test all Op codes (except /47/) for a valid GR address (R1 address) in DR-T because a GR can only be addressed by an address higher than /7/.

   If DR-T does not contain an 8 bit, the CPU stops and DR-R contains a /5/ to indicate an Address error (Figure 3-27a).
4. The Op Code and Byte 2, located in DR-E, S, T, and R, are stored in the Op Reg. and Byte 2 Reg. in AUX-storage positions /00 and 01/ for later use. (Subroutine ESTR Reg)
5. An address exchange moves the I address of byte 3 in the instruction, from the Subregister into DR-A, S, T, and R.
6. The Subroutine "Storage to ESTR" moves bytes 3 and 4 of the instruction from Main storage into DR-E, S, T, and R, following which DR-A, S, T, and R contain the NSI address, DR-E contains the B2 field of the instruction and Subregisters S, T, and R contain the D2 field of the instruction.
7. An address exchange moves the B2 and D2 field into DR-(E) S, T, and R and the NSI address into the Subregisters.
8. The B2 field in DR-E is tested for an 8 bit to determine if indexing is required. If DR-E contains an 8 bit, indexing is required and a GR is addressed. The contents of the GR is added to the D2 field located in DR-S, T, and R by the Indexing routine. The indexing routine is described in the section on Indexing.

   NOTE: DR-A, S, T and R now contains the Operand 2 address. The Operand 2 address is specified by the B2 and D2 field, as is, when no indexing is required, or by the result of the Indexing routine when indexing is required.
9. The Operand 2 address is tested for validity. In a 4K storage, addresses up to 4096 are valid; in an 8K storage addresses up to 8092

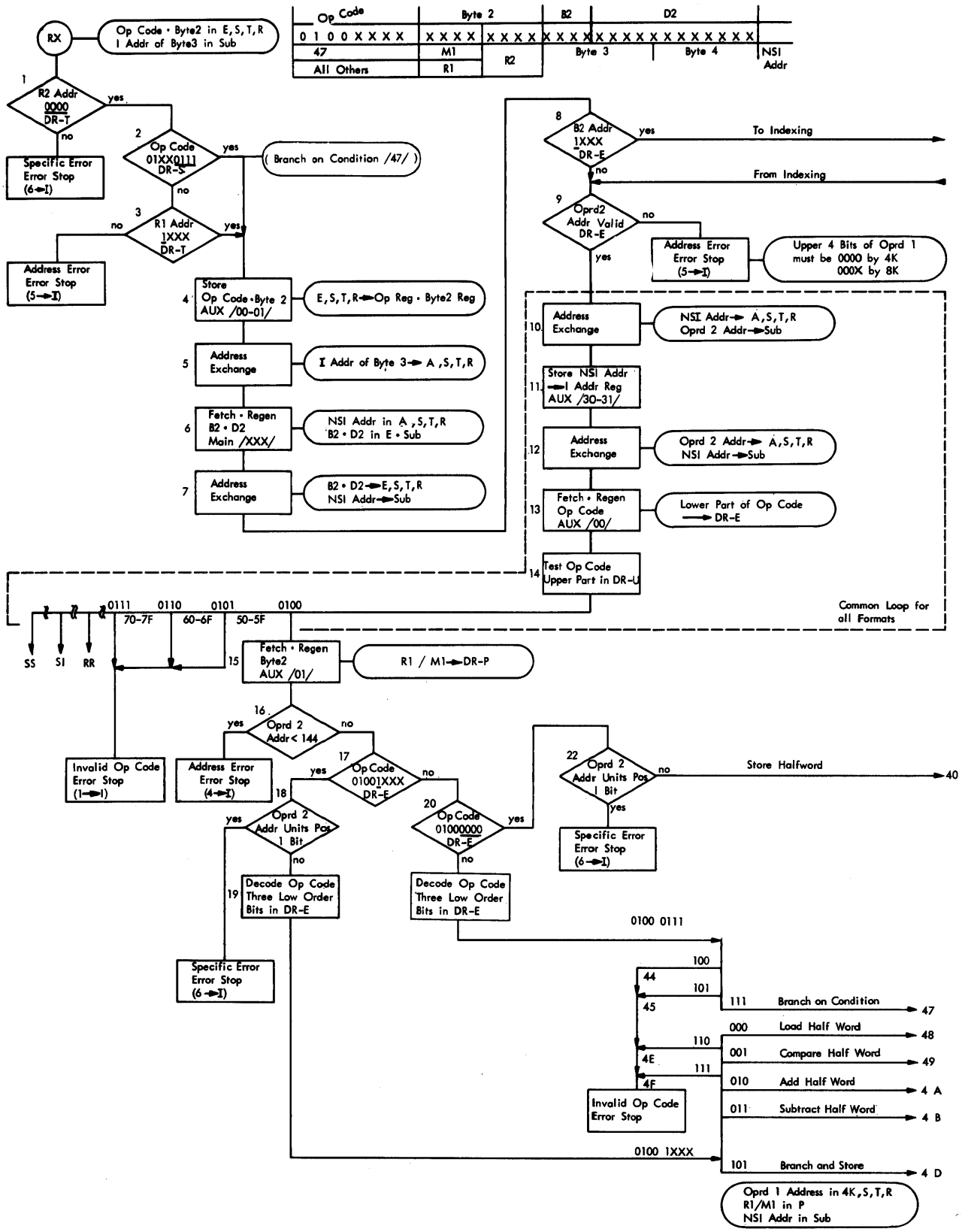


Figure 3-32 I-phase Op Decoding RX

are valid; in a 16K storage, addresses up to 16,384 are valid.

DR-E contains the 4 high order address bits of the Operand 2 address and should be 0000 for a 4K storage (Addresses up to 4096); 000X for an 8K storage (Addresses up to 8192); and 00XX for a 16K storage (addresses up to 16,384).

If the Operand 2 address is not greater than the storage size, continue as shown in step 10. If the Operand 2 address is greater than the storage size, the CPU stops and DR-I contains a /5/ to indicate an Address error (Figure 3-27a).

10. An address exchange moves the NSI address into DR- A, S, T, and R, and moves the Operand 2 address into the Subregister.
11. The contents of DR-A, S, T, and R (NSI Address) is stored (Subroutine STR to Reg) in the I Address Reg, AUX storage positions /30 and 31/ to save it for the next I-phase.
12. An address exchange moves the Operand 2 address into DR- A, S, T, and R, and moves the NSI address into the Subregisters.
13. The Op code is read out of the Op Reg (AUX storage position /00/) into DR-U and L. The contents of DR-L (lower part of Op code) is moved into DR-E for later use.
14. The contents of DR-U (upper part of Op code) is tested by a USE DR-U for further decoding.

    If the contents of DR-U is 0101 (/5/), 0110 (/6/), or 0111 (/7/), the Op code is invalid and the CPU stops with a /1/ in DR-I to indicate an Invalid Op code (Figure 3-27a). If the contents of DR-U is 0100 (/4/), continue as shown in step 15.
15. Byte 2 of the instruction is read out of the Byte 2 Reg (AUX storage position /01/) into DR-U and L. The contents of DR-U (R1/M1 field) is moved into DR-P for later use.
16. The Operand 2 address is tested to see that it does not address the protected area in Main storage (000 through 143). The subroutine "Storage Protect" tests the contents of A, S, T, and R for an address less than 144 (/090/). If the Operand 2 address is less than 144, the CPU stops with a /4/ in DR-I to indicate an Address error (Figure 3-27a).

    If the Operand 2 address is equal to, or greater than 144, continue as shown in step 17.
17. Test the contents of DR-E for further Op decoding. If DR-E contains an 8 bit, continue as shown in step 18. If DR-E does not contain an 8 bit, continue as shown in step 20.
18. Test the Operand 2 address for "Boundary" (even Main storage address). All Op codes in which one address refers to a GR, and the other address to the Main storage, must address an even (Boundary) Main storage position.

    If DR-R contains a 1 bit, the Operand 2 address is odd and the CPU stops with a /6/ in DR-L to indicate a Specification error (Figure 3-27a). If DR-R does not contain a 1 bit, the Operand 2 address is even (correct). Continue as shown in step 19.
19. The Op codes are decoded depending upon the three low order bits of the Op code in DR-E (position 4, 2, and 1). Position 8 of DR-E is decoded as shown in step 17. The following table shows the contents of DR-E and the equivalent Op code.

DR-E Content	Op Code
P 8 4 2 1	
0 1 0 0 0	/48/
1 1 0 0 1	/49/
1 1 0 1 0	/4A/
0 1 0 1 1	/4B/
0 1 1 0 1	/4D/
0 1 1 1 0	Invalid
1 1 1 1 1	Invalid

When a valid Op code is detected (decoded) the associated E-phase is executed.

When an invalid Op code is detected the CPU stops with a /1/ in DR-I to indicate an Invalid Op code (Figure 3-27a).

20. The contents of DR-E are tested for being /0/. If DR-E contains a /0/ (Op code /40/ Store Half word is decoded), continue as shown in step 22. If DR-E does not contain /0/ (further decoding required), continue as shown in step 21.
21. The Op codes are decoded depending upon the three low order bits of the Op code in DR-E (position 4, 2, and 1). Position 8 of DR-E is decoded as shown in step 17. The following shows the contents of DR-E and the equivalent Op Code.

DR-E Content	Op Code
P 8 4 2 1	
0 0 1 1 1	/47/
0 0 1 0 0	Invalid
1 0 1 0 1	Invalid

When a valid Op code is detected (decoded), the associated E-phase is executed.

When an invalid Op code is detected, the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

22. Test Op code /40/ for correct "Boundary" address. If DR-R contains a 1 bit, the operand 2 address is odd and the CPU stops. DR-I contains a /6/ to indicate a Specification error (Figure 3-27a). If DR-R contains no 1 bit, the Operand 2 address is even (correct) and the E-phase of Op code /40/ is executed.

SI Format Op Code Decoding

- There are 10 valid Op codes in the SI Format.
- The SI Format is already decoded in the previous part of the I-phase (Figure 3-29).

Description (Figure 3-33)

1. The Op Code and Byte 2 are located in DR-E, S, T, and R. The Op code is stored in the Op Register. Byte 2 is stored in the Byte 2 Register in AUX-storage positions /00-01/ for later use.
2. An address exchange moves the I-address of byte 3 in the instruction from the Sub-register into DR-A, S, T, and R.
3. B1 and D1 values of the instruction are read out of Main storage into DR-E, S, T, and R by the Stor to ESTR subroutine.

   After completion of the Stor to ESTR subroutine, B1 and D1 values are located in DR-E and the sub-register. The NSI address is in DR- A, S, T, and R.
4. An address exchange moves the NSI address into the sub-register and the B1 and D1 values into DR-E, S, T, and R.
5. The B1 field, (located in DR-E), is tested for an 8-bit to determine if indexing is required.

   If DR-E contains an 8-bit, indexing is required and a GR is addressed. The contents of the GR is added to the D1 field located in DR-S, T, and R by the indexing routine.

   If DR-E does not contain an 8-bit, no indexing is required. Continue as shown in step 6.

   NOTE: DR- A, S, T, and R now contain the Operand 1 address. The Operand 1 address is specified by the B1 and D1 field, as is, when no indexing is required, or by the result of the indexing routine when indexing is required.
6. The Operand 1 address is tested for validity. DR-E contains the four high order bits of the Operand 1 address. The bit patterns in DR-E should be as follows:

Storage	Bit Pattern
4K	Not greater than 0000
8K	Not greater than 0001
16K	Not greater than 0010

   If the contents of DR-E are higher than shown above, the Operand 1 address is invalid and the CPU stops. DR-I contains a /5/ to indicate an address error (Figure 3-27a).
7. An address exchange moves the NSI address into DR- A, S, T, and R and the Operand 1 address into the sub-register.
8. The content of A , S, T, and R (NSI address) is stored (STR to Reg subroutine) into the I-address Register (AUX-Storage position /30-31/) to save it for the next I-phase.
9. An address exchange moves the Operand 1 address into DR- A, S, T, and R.
10. The Op code is read out of the Op Reg (AUX-Storage position /00/) into DR-U and L. The contents of DR-L (lower part of the Op code) is moved into DR-E for later use.
11. The contents of DR-U (upper part of Op code) is tested by a USE DR-U for further decoding. If the contents of DR-U is 1011(/B/) or

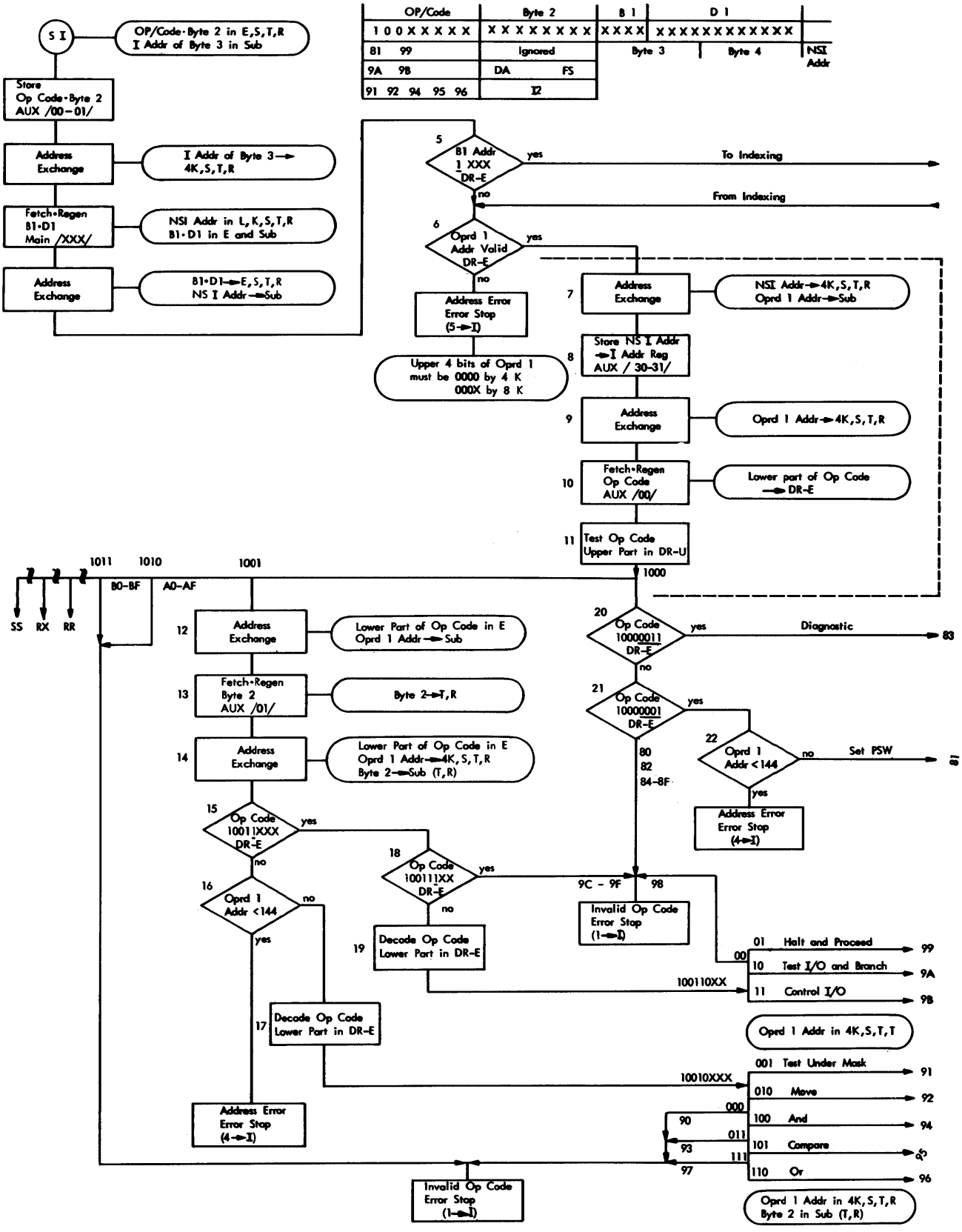


Figure 3-33 I-phase Op Decoding SI

1010 (/A/), the Op code is invalid and the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

If the contents of DR-U is 1001 (/9/), continue as shown in step 12.

If the contents of DR-U is 1000(/8/), continue as shown in step 20.

12. An address exchange moves the Operand 1 address into (DR-E) sub-register.
13. Byte 2 of the instruction is read out of the Byte 2 Register in AUX-Storage position /01/ and is moved into DR-T, and R for later use in the E-phases.
14. An address exchange moves the contents of the DR's into the sub-register and vice versa.

NOTE: After the exchange, the registers contain the following:

Register	Contents
DR-E	Lower part of the Op code
DR-A, S, T, and R	Operand 1 address
Sub-registers T and R	Byte 2 of the instruction

15. Test the contents of DR-E for further Op decoding.

If DR-E contains an 8-bit, continue as shown in step 18.

If DR-E contains no 8-bit, continue as shown in step 16.

16. The Operand 1 address is tested to see that it does not address the protected area in Main Storage (000 through 143). The Storage Protect subroutine tests the contents of DR-A, S, T, and R for an address less than 144 (/90/). If the Operand 2 address is less than 144, the CPU stops with a /4/ in DR-I to indicate an address error (Figure 3-27a).

If the Operand 1 address is equal to or greater than 144, continue as shown in step 17.

The Op codes are decoded depending upon the three low order bits of the Op code in DR-E (position 4, 2, and 1). Position 8 of DR-E is decoded as shown in step 15. The contents of DR-E and the equivalent Op code is shown below:

DR-E Content	Op Code
P 8 4 2 1	
0 0 0 0 1	/91/
0 0 0 1 0	/92/
0 0 1 0 0	/94/
1 0 1 0 1	/95/
1 0 1 1 0	/96/
1 0 0 0 0	Invalid
1 0 0 1 1	Invalid
0 0 1 1 1	Invalid

When a valid Op code is detected, (decoded) the associated E-phase is executed.

When an invalid Op code is detected, the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

18. Test the contents of DR-E for further Op decoding.

If DR-E contains a 4-bit, an invalid Op code is detected and the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

If DR-E contains no 4-bit, continue as shown in step 19.

19. The Op codes are decoded depending upon the two low order bits of the Op code in DR-E (position 2 and 1). Position 8 and 4 of DR-E are decoded as shown in steps 15 and 18. The contents of DR-E and the equivalent Op code is shown below:

DR-E Content	Op Code
P 8 4 2 1	
1 0 0 0 1	/99/
1 1 0 1 0	/9A/
0 1 0 1 1	/9B/
0 1 0 0 0	Invalid

When a valid Op code is detected (decoded), the associated E-phase is executed.

When an invalid Op code is detected, the CPU stops with a /1/ in DR-T to indicate an invalid Op code (Figure 149A).

20. Test the Op code in DR-E for further Op decoding. If DR-E contains 0011, the Op code /83/ (diagnostic) is determined and the E-phase of it is executed.

If DR-E does not contain 0011, continue as shown in step 21.

21. Test the Op code in DR-E for farther Op decoding.

    If DR-E contains 0001, continue as shown in step 22.

    If DR-E does not contain 0001, an invalid Op code is detected and the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

22. The Operand 1 address is tested to see that it does not address the protected area in Main Storage (000 through 143). The Storage Protect subroutine tests the contents of DR-A, S, T, and R for an address less than 144 (/90/).

    If the Operand 1 address is less than 144, the CPU stops with a /4/ in DR-I to indicate an address error (Figure 149A).

    If the Operand 1 address is equal to or greater than 144, Op code /81/ (Set PSW) is decoded and its E-phase is executed.

SS Format Op Code Decoding

- There are 15 valid Op codes in the SS Format.
- The SS Format is already decoded in the previous part of the I-phase (Figure 3-29).

Description (Figures 3-34 and 3-34a)

1. The Op code and Byte 2 are located in DR-E, S, T, and R. The Op code is stored in the Op Register. Byte 2 is stored in the Byte 2 Register in AUX-Storage positions /00-10/ for later use.
2. An address exchange moves the address of byte 3 in the instruction from the sub-register into A, S, T, and R.
3. B1 and D1 values of the instruction are read out of Main Storage into DR-E, S, T, and R by the Memto ESTR subroutine.

    After completion of the Memto ESTR subroutine, the B1 and D1 values are located in DR-E and the sub-register. The address of byte 5 is in DR-A, S, T, and R.
4. An address exchange moves the B1 and D1 values into DR-E, S, T, and R and the address of byte 5 of the instruction into the sub-register.
5. The B1 field located in DR-E is tested for an 8-bit to determine if indexing is required.

    If DR-E contains an 8-bit, indexing is required and a GR is addressed. The contents of the GR are added to the D1 field located in DR-S, T, and R by the indexing routine.

    If DR-E does not contain an 8-bit, no indexing is required. Continue as shown in Step 6.

    NOTE: DR- A, S, T, and R now contains the Operand 1 address. The Operand 1 address is specified by the B1 and D1 field, as is, when no indexing is required, or by the result of the indexing routine when indexing is required.

6. The Operand 1 address is tested for validity. DR-E contains the four high order bits of the Operand 1 address. The bit patterns in DR-E should be:

Storage	Bit Pattern
4K	Not greater than 0000
8K	Not greater than 0001
16K	Not greater than 0010

    If the contents of DR-E is not greater than shown above, continue with step 7.

    If the contents of DR-E is greater than shown, the Operand 1 address is invalid and the CPU stops. DR-I contains a /5/ to indicate an address error (Figure 3-27a).
7. The contents of A, S, T, R (Operand 1 address) is stored (Subroutine STR to REG) into the Operand 1 Address Register AUX-Storage position /40-41/ to save it for later use in the E-phase.
8. An address exchange moves the address of byte 5 into DR-A, S, T, and R.
9. B2 and D2 values of the instruction are read out of Main Storage into DR-E, S, T, and R by the Stor to ESTR subroutine.

    After completion of the Stor to ESTR subroutine, the B2 and D2 values are located in DR-E and the sub-register. The NSI-address is in DR- A., S, T, and R.
10. An address exchange moves the B2 and D2 values into DR-E, S, T, and R and the NSI-address into the sub-register.
11. The B2 field located in DR-E is tested for an 8-bit to determine if indexing is required.

Op Code	Byte 2		B1	D1	B2	D2	
11XXXXXX	XXXXXXXX		XXXX	XXXXXXXXXXX	XXXX	XXXXXXXXXXX	
D1,2,3,5,E,C	L		Byte 3	Byte 4	Byte 5	Byte 6	NSI Addr
F1,2,3,8,9,A,B,C	L1	L2					
D0	DA	FS					

FIELD LENGTH

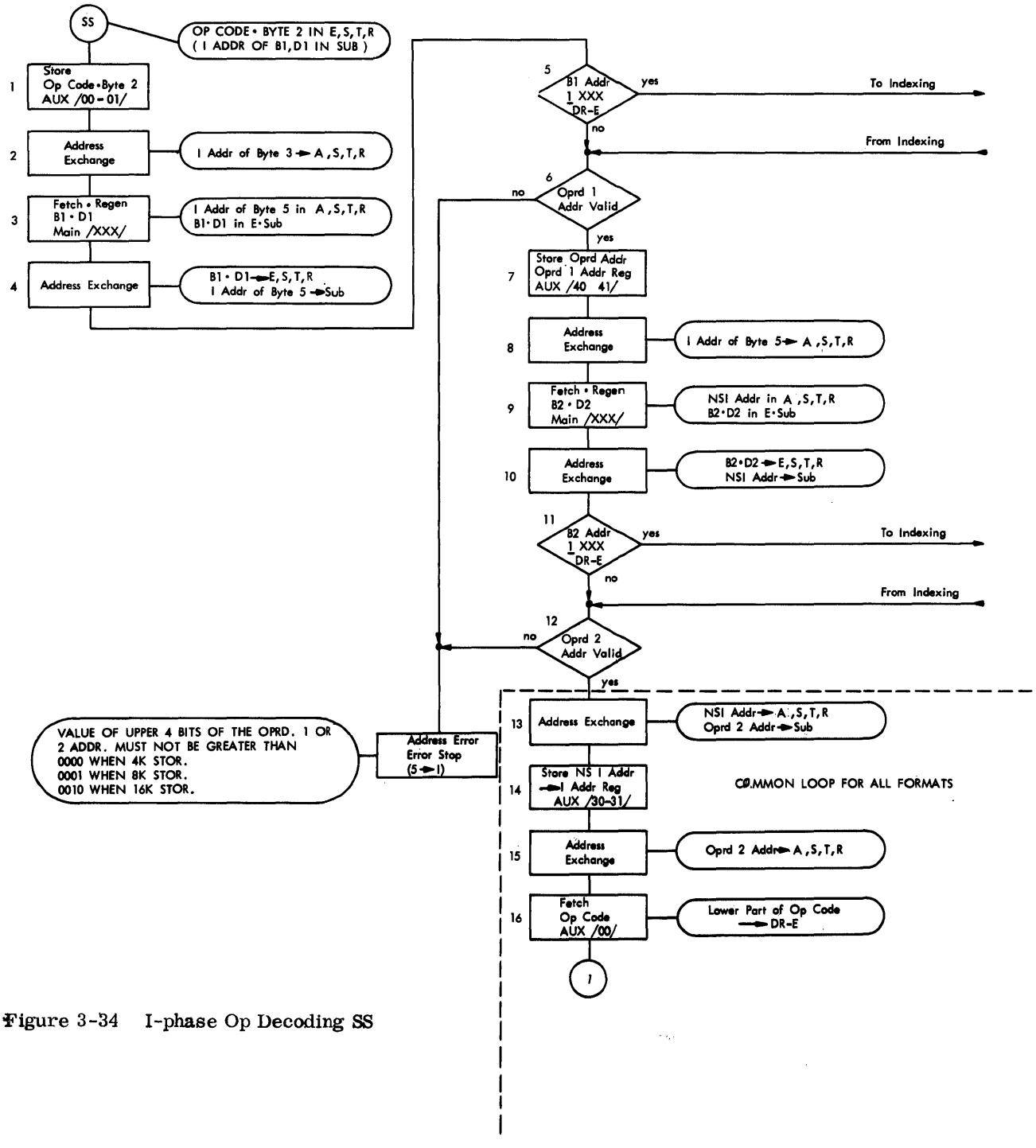


Figure 3-34 I-phase Op Decoding SS

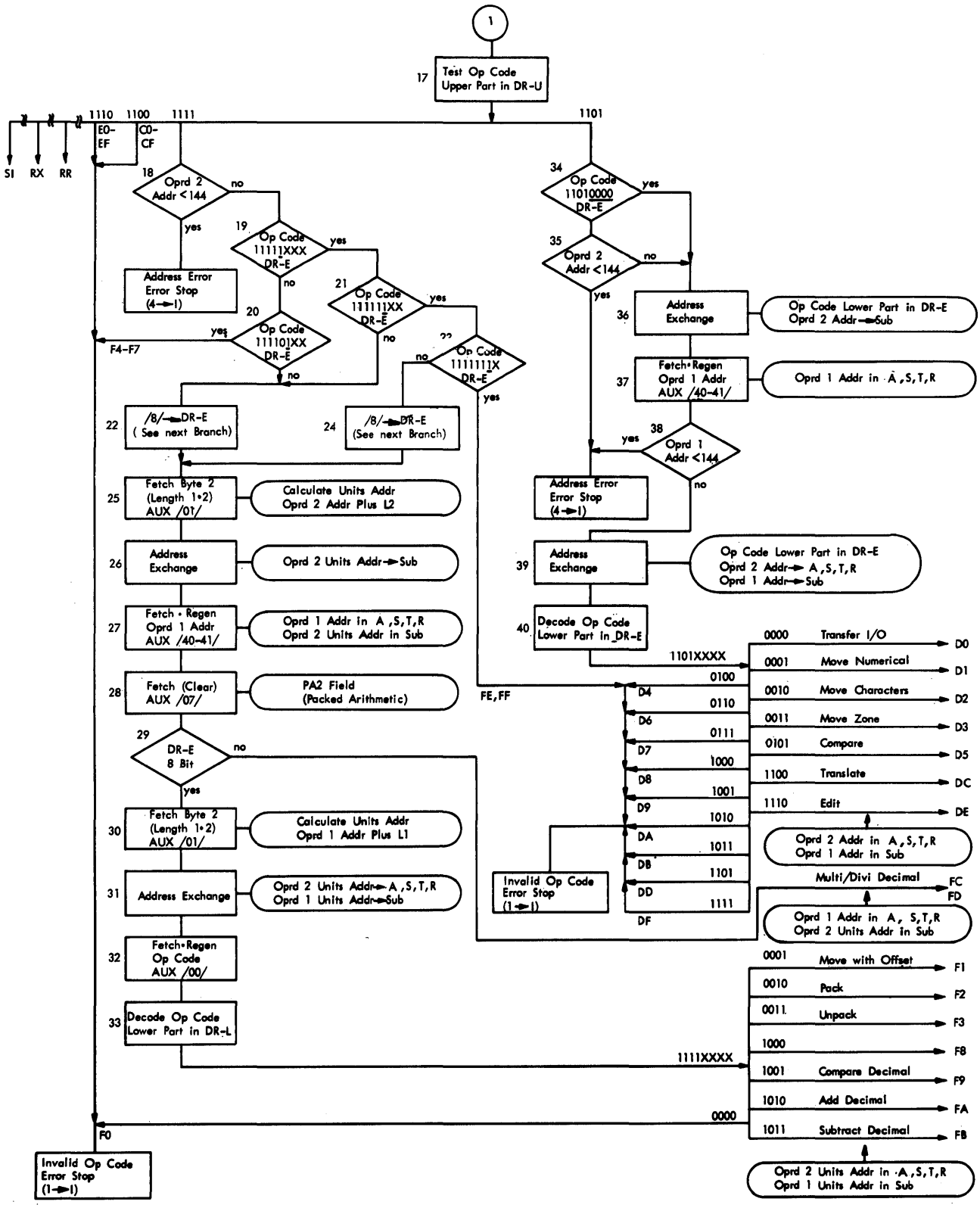


Figure 3-34a I-phase Op Decoding SS

If DR-E contains an 8-bit, indexing is required and a GR is addressed. The contents of the GR are added to the D2 field located in DR-S, T, and R by the indexing routine.

If DR-E does not contain an 8-bit, no indexing is required. Continue as shown in step 12.

NOTE: DR- A, S, T, and R now contain the Operand 2 address. The Operand 2 address is specified by the B2 and D2 field, as is, when no indexing is required, or by the result of the indexing routine when indexing is required.

12. The Operand 2 address is tested for validity. DR-E contains the four high order bits of the Operand 2 address. The bit pattern in DR-E is as follows:

Storage	Bit Pattern
4K	Not greater than 0000
8K	Not greater than 0001
16K	Not greater than 0010

If the contents of DR-E is not greater than shown above, continue with step 13.

If the contents of DR-E is greater than shown above, the Operand 2 address is invalid and the CPU stops. DR-I contains a /5/ to indicate an address error (Figure 3-27a).

13. An address exchange moves the NSI-address into A, S, T, R and the Operand 2 address into the sub-registers.
14. The contents of A , S, T, and R (NSI-address) is stored (STR to Reg subroutine) into I Address Register (AUX-Storage positions /30-31/) to save it for the next I-phase.
15. An address exchange moves the Operand 2 address into A, S, T, and R.
16. The Op code is read out of the Op Reg (AUX-Storage position /00/) into DR-U and L. The contents of DR-L (lower part of the Op code) is moved into DR-E for later use.
17. The contents of DR-U (upper part of Op code) is tested by a USE DR-U for further decoding. If the contents of DR-U is 1110 (/E/) or 1100 (/C/), the Op code is invalid and the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

If the contents of DR-U is 1111 (/F/), continue as shown in step 18.

If the contents of DR-U is 1101 (/D/), continue as shown in step 33.

18. The Operand 2 address is tested to see that it does not address the protected area in Main Storage (000 through 143). The Storage Protect subroutine tests the content of A, S, T, and R for an address less than 144 (/90/).

If the Operand 2 address is less than 144, the CPU stops with a /4/ in DR-I to indicate an address error (Figure 3-27a).

If the Operand 2 address is equal to or greater than 144, continue as shown in step 19.

19. Test the contents of DR-E for further Op decoding.

If DR-E contains an 8-bit, continue as shown in step 21.

If DR-E contains no 8-bit, continue as shown in step 20.

20. Test the contents of DR-E for further Op decoding.

If DR-E contains no 4-bit (and no 8-bit), continue as shown in step 22.

If DR-E contains a 4-bit (and no 8-bit), an invalid Op code is detected.

The CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

21. Test the contents of DR-E for further Op decoding.

If DR-E contains no 4-bit (and an 8-bit), continue as shown in step 22.

If DR-E contains a 4-bit (and an 8-bit), continue as shown in step 23.

22. An /8/ is moved into DR-E as indicated for further decoding. Continue as shown in step 25.
23. Test the contents of DR-E for further Op decoding.

If DR-E contains no 2-bit (and an 8-bit and a 4-bit), continue as shown in step 24.

If DR-E contains a 2-bit (and an 8 and a 6-bit), an invalid Op code is detected and the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

24. A /0/ is moved into DR-E as an indicator for further decoding. Continue as shown in step 25.
25. Byte 2 is read out of the Byte 2 Register (AUX-Storage position /01/) into DR-U and L.

DR-L contains the lower part of Byte 2, which represents the L2 (Length 2) field of the instruction. The subroutine "Calculate Units Address" adds the Operand 2 address and the L2 field together to generate the units address of Operand 2, which is used in the E-phase of the decoded Op code.

26. An address exchange moves the Operand 2 units address into the Sub-register.
27. The Operand 1 address is read out of the Operand 1 Address Register (AUX-Storage position /40-41/) into DR- A , S, T, and R for later use.
28. The PA2 Register (AUX-Storage position /07/) is read out, to clear this position for later use in the various E-phases.
29. Test the contents of DR-E for further Op decoding.

    If DR-E contains an 8-bit, continue as shown in step 30.

    If DR-E contains no 8-bit, an Op code Multiply Decimal (/FC/) or Divide Decimal (/FD/) is determined and the Multi Div Decimal E-phase is executed.
30. Byte 2 is read out of the Byte Register (AUX-Storage position /01/) into DR-U and L. DR-U contains the upper part of byte 2, which represents the L1 (Length 1) field of the instruction. The Calculate Units Address subroutine adds the Operand 1 address and the L1 field together to generate the units address of Operand 1 which is used in the E-phase of the decoded Op code.
31. An address exchange moves the Operand 2 units address into DR-A , S, T, and R and the Operand 1 units address into the sub-register.
32. The Op code is read out of the Op Register (AUX-Storage position /00/) into DR-U and L.
33. The contents of DR-L (lower part of Op code) is tested by a USE DR-L for further decoding. The following chart shows the contents of DR-L and the equivalent Op code.

DR-L P 8 4 2 1	Op Code
0 0 0 0 1	/F1/
0 0 0 1 0	/F2/
1 0 0 1 1	/F3/
0 1 0 0 0	/F8/
1 1 0 0 1	/F9/
1 1 0 1 0	/FA/
0 1 0 1 1	/FB/
1 0 0 0 0	Invalid

When a valid Op code is detected (decoded), the associated E-phase is executed.

When an invalid Op code is detected, the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

34. Test the contents of DR-E for further Op decoding.

    If the contents of DR-E is 0000 (Op code Transfer I/O), continue as shown in step 36. because, for a Transfer I/O Op code, the Operand 2 address field contains the field length values.

    If the contents of DR-E is not 0000, continue as shown in step 35.
35. The Operand 2 address is tested to see that it does not address the protected area in Main Storage (000 through 143). The Storage Protect subroutine tests the contents of 4K, S, T, and R for an address less than 144 (/90/).

    If the Operand 2 address is less than 144, the CPU stops with a /4/ in DR-I to indicate an address error (Figure 3-27a).

    If the Operand 2 address is equal to or greater than 144, continue as shown in step 36.
36. An address exchange moves the Operand 2 address into the sub-register.

    The lower part of the Op code is in DR-E.
37. The Operand 1 address is read out of the Operand 1 Address Register into DR- A , S, T and R for test purposes.
38. The Operand 1 address is tested to verify that it does not address the protected area in Main Storage (000 through 143). The Storage Protect subroutine tests the contents of A , S, T, and R for an address less than 144 (/90/).

    If the Operand 1 address is less than 144, the CPU stops with a /4/ in DR-I to indicate an address error (Figure 3-27a).

    If the Operand 1 address is equal to or greater than 144, continue as shown in step 39.
39. An address exchange moves the Operand 2 address into A , S, T, and R and the Operand 1 address into the sub-register. DR-E contains the lower part of the Op code.
40. The contents of DR-E (lower part of the Op code) is tested by a USE DR-E for further decoding. The contents of DR-E and the equivalent Op codes are shown below:

DR-E P 8 4 2 1	Op Code
1 0 0 0 0	/D0/
0 0 0 0 1	/D1/
0 0 0 1 0	/D2/

1 0 0 1 1	/D3/
1 0 1 0 1	/D5/
1 1 1 0 0	/DC/
0 1 1 1 0	/DE/
0 0 1 0 0	Invalid
1 0 1 1 0	Invalid
0 0 1 1 1	Invalid
0 1 0 0 0	Invalid
1 1 0 0 1	Invalid
1 1 0 1 0	Invalid
0 1 0 1 1	Invalid
0 1 1 0 1	Invalid
1 1 1 1 1	Invalid

When a valid Op code is detected (decoded), the associated E-phase is executed.

When an invalid Op code is detected, the CPU stops with a /1/ in DR-I to indicate an invalid Op code (Figure 3-27a).

Indexing

- Indexing is a micro-program routine within the I-phase micro-program.
- Objective: Generate an effective main storage address depending upon the contents of a GR addressed by the Base (B) address and the Displacement (D) bits of a machine instruction.
- Indexing is required when the B address refers to a GR (1XXX).
- The B address is a 4 bit address for addressing a GR.
- D is a 12 bit field which, for indexing, is added to the contents of the GR addressed by the B address.
- The result of the addition is the effective or the Operand (1 or 2) address.
- A machine instruction can have one or two B addresses and D fields (Figure 3-35).

Description (Figure 3-36 and 3-37)

Figure 3-36 illustrates where the B1/B2 and D1/D2 fields are located, before the Index routine and where the Operand 1/ Operand 2 addresses are located after the routine.

Prerequisite. The Indexing routine is entered from the I-phase only (Figures 3-36 and 3-37). For purposes of this description the contents of the GR addressed by B1 or B2 is labeled "Y" and D1 or D2 addresses are labeled "X" in Figure 3-36 and 3-37

1. Y2 · Y1 (the low order byte in the GR addressed by B1 or B2) is read out into DR-U and L. It is then stored back (regenerated) into the GR.
2. The X1 and Y1 addresses are added together by the Address Subroutine to develop Result 1 (X1 + Y1) which is then stored in DR-R.
3. X2 and Y2 are added together by the Addr Subroutine to give Result 2 (X2 + Y2) which is then stored in DR-T.
4. Y4 · Y3 (the left byte in GR addressed by B1 or B2) are read into DR-U and L. It is then stored back (regenerated) into the GR.
5. X3 and Y3 are added together by the Addr Subroutine to develop Result 3 which is then stored in DR-S.
6. A Test is made of DR-E to determine if a carry resulted from the previous add operation. DR-E contains a 1 bit if a carry was generated. Continue as shown in step 7 if a carry was generated. Continue as shown in step 8 if a carry was not generated.
7. When DR-E contains a 1 bit (carry of Result 3) a 1 is added to Y4 to generate Result 4.
   a. Y4 should be 0000 if the CPU has a 4K storage.
   b. Y4 should be 000X if the CPU has an 8K storage.
   c. Y4 should not be greater than 0010 if the CPU has a 16K storage.
   d. Result 4 should be 0000 if the CPU has a 4K storage.
   e. Result 4 should be 0001 if the CPU has an 8K storage.
   f. If Result 4 is greater than 0000 (4K) 0001 (8K) an Address error occurs. This error is detected next step after the Indexing routine re-enters the I-phase.
8. For an 8K storage, the right most position of Result 4 (000X) is moved into DR-A.
9. Move Result 4 into DR-E to have the complete Operand 1 or Operand 2 address in E, S, T, R for display if an error (invalid Operand 1 or 2 address) stop occurs. The normal I-phase is resumed.

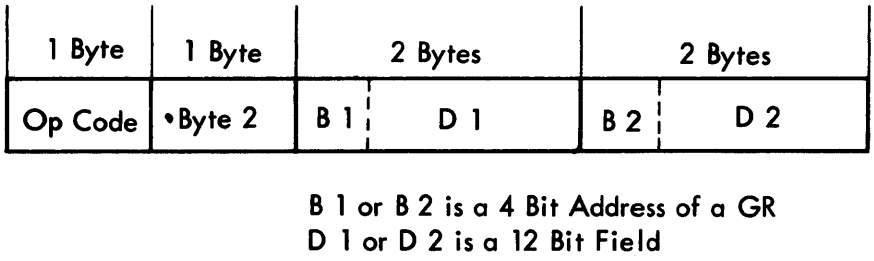


Figure 3-35 Bl/B2 and Dl/D2 Field Locations

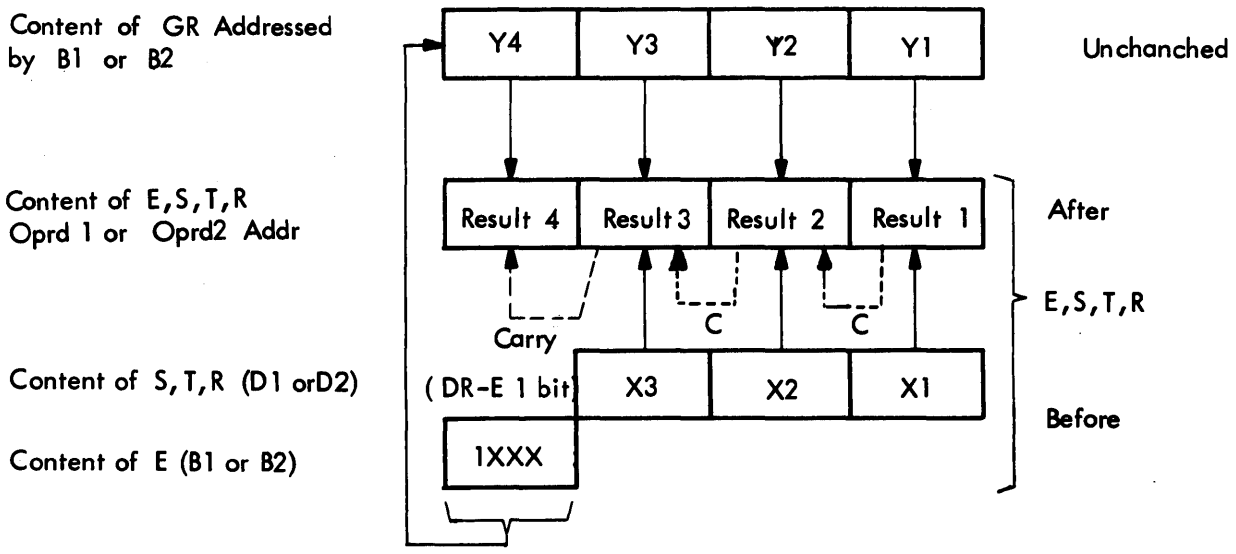


Figure 3-36 Indexing Location of Values

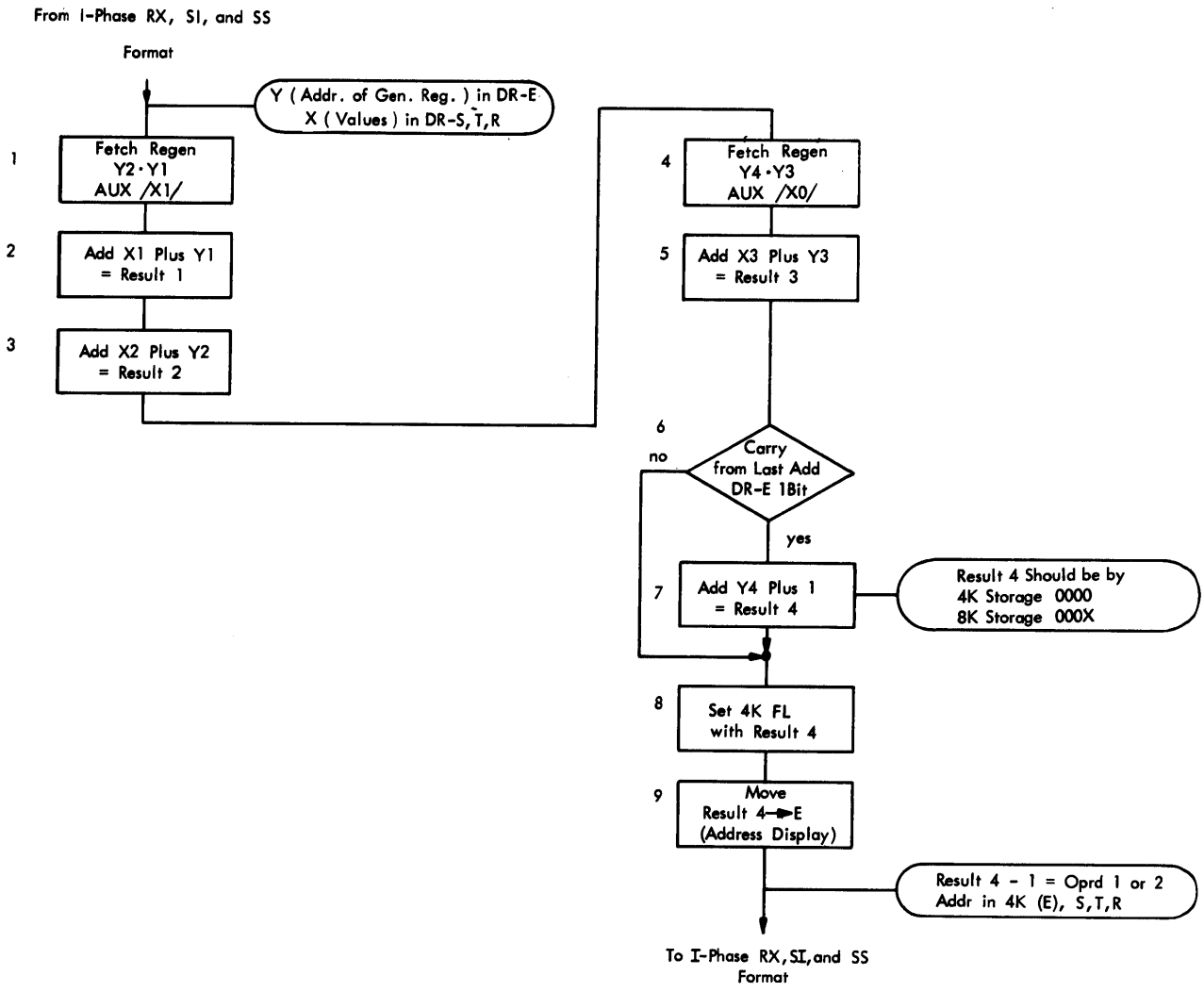


Figure 3-37 Indexing Flow Chart

Interrupt

- Objective: The Interrupt tells the CPU that an I/O unit has just ended its current operation. This enables the CPU to immediately use another I/O device without wasting time. Unneccessary idle time of I/O devices is thus avoided.
- The Interrupt is a micro-program within the I-phase.
- The Interrupt is only performed when 2 conditions are satisfied: The channel mask of the PSW must contain a 1-bit and there must be an Interrupt bit set in one of the Interrupt bytes in Auxiliary Storage.
- The Interrupt can be disabled by setting the 1-bit in the channel mask to zero.
- Interrupt bits are set into the Auxiliary Storage when an I/O device has a channel end condition.
- The channel end condition is defined as that time in the mechanical cycle of the device at which the data transfer has been completed.
- If more than one I/O unit has reached the channel end condition, the various channel end conditions are tested step by step according to a preestablished priority sequence (Figure 3-38).
- The first channel end condition encountered in this sequence causes the interrupt to occur.
- The Interrupt is accomplished by replacing the current PSW with a new one which is fetched from a fixed Main Storage position.
- Due to the new NSI address, the Interrupt is actually an automatic branch in the machine program.
- After the branch program has been completely processed, a Set PSW instruction may bring back the old PSW and the machine program returns to the point at which it was interrupted if so desired.

Description

Data processing is accomplished in a specific sequence of operations: Manual Routines, Interrupt, I-phase and E-phase (See Figure 3-29, Step 48).

48. After the manual routines, the micro-program fetches the channel mask to see whether or not a channel mask bit is set.
49. If there is no 1-bit in DR-L, the interrupt is completely ignored because it is disabled. The micro-program continues in the I-phase. However, if there is a 1-bit in DR-L, the Interrupt bytes in Auxiliary Storage are fetched to see whether they contain any bits (68,69).
70. When an I/O unit comes to a channel end condition, an interrupt bit is set in one of the 2 Interrupt bytes. This occurred some time before. Now, a test of these bits causes either an interrupt or, depending on the result, allows the micro-program to continue in the I-phase if no bit was turned on.

    The interrupt routine is entered (Figure 3-39) when there were bits present in the Interrupt bytes (-4-). At this point, the NSI Address (Of the current PSW) is in DR-E, S, T, R. Normally, this would be the next instruction. The interrupt, however, represents a branch in the program. Therefore it is necessary to exchange the current PSW for a new one with a different NSI.

1. A 2 bit is set into DR-P. This bit is an indicator which tells the micro-program whether or not to repeat a specific series of micro-steps.
2. The address of the right-most byte of the old PSW is generated in DR-S, T, R. It is evident that prior to this address generation, the content of DR-E, S, T, R (NSI of current PSW) was set into the Sub-registers. This is not shown in Figure 149c.
3. The address just generated (/093/) is used to read out the position /093/ of Main Storage. Since the current PSW is to be stored in the old PSW area, this area must be cleared first.
4. Half bytes Y2 and Y1 of the current PSW are set into DR-U and L so they can later

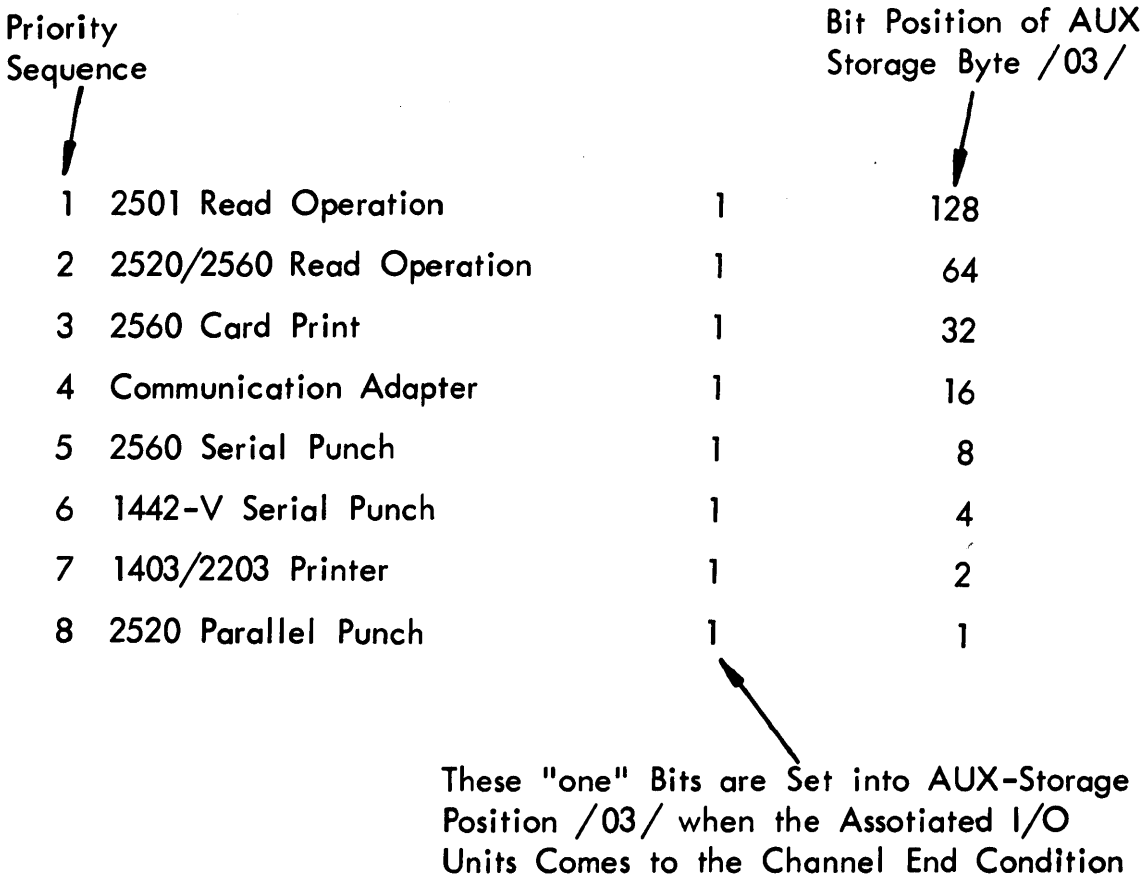


Figure 3-38 Interrupt Priority Sequence

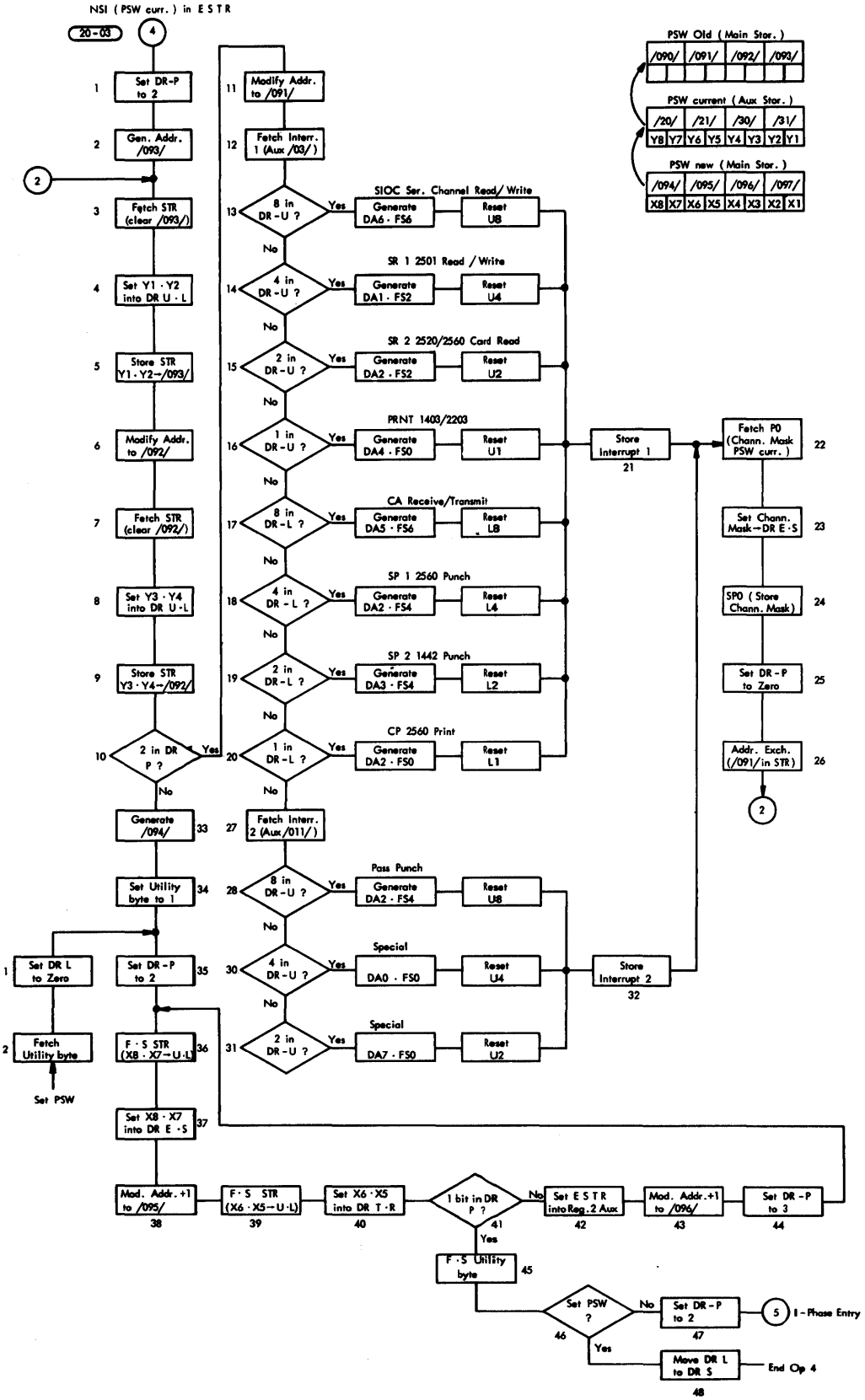


Figure 3-39 Interrupt

be stored into the old PSW area. Prior to this, however, a Control 15 operation brought back the NSI address of the current PSW (Y4, Y3, Y2, Yl) from the Sub-register. This is not shown.

5. Since Y2 and Yl were moved from DR-T and R into DR-U and L, another Control 15 (not shown) brings back address /093/ from the Sub-registers into DR-S, T, R. This address is used to store Yl and Yl into the old PSW.
6. The address /093/ is modified by the R-1 Sub-routine to /092/.
7. The address just modified is used to read out (clear) the next byte of the old PSW.
8. A Control 15 operation (not shown) again puts the NSI address of the current PSW into DR-E, S, T, R. The DR-E, S, T, R now contains Y4, Y3, Y2, Yl. The half-bytes Y4 and Y3 are moved into DR-U and L, as before, because fetching and storing of information can only be accomplished by way of DR-U and L.
9. The content of DR-U and L (Y4 Y3) is stored into main storage position /092/. With this operation, half of the current PSW is stored in the old PSW area.
10. A test is performed to determine whether the indicator bit in DR-P is On.
11. Since there was a 2-bit in DR-P, the micro-program continues with a R-1 operation. This modifies the address to /091/.
12. The modified address /091/ is set into the Sub-registers by a Control 15 (not shown) and then the interrupt byte 1 (Auxiliary position /04/) is fetched.
13. A total of 9 interrupt bits are possible. The micro-program checks for these bits step by step according to the priority sequence. The priority sequence cannot be changed as it is predetermined by the designer. If there was no 8-bit on, that is Serial I/O Channel Read or Write, the next bit (Serial Read or Write 250l) is tested and so on. When an interrupt bit is found, for example, bit 2 in DR-U, it is recognized that either the 2520 or the 2560 (whichever is installed) requests service for a Card Read operation.
15. The Device Address and the Function Specification are generated in DR-T and R respectively. This is necessary to be able to store the Device Address and the Function Specification into the old PSW. This provides a means to find out later which device caused the interrupt and with which operation.
21. The interrupt bit is reset and the interrupt byte is stored back into the Auxiliary Storage position /20/.
22. The left-most byte of the current PSW which contains the channel mask and the condition code is fetched from the Auxiliary Storage and set into DR-U and L so that it can be stored into the old PSW in a later step.
23. The content of DR-U, L is moved to DR-E, S. With this operation, the second half of the current PSW is finally located in DR-E, S, T, R. The micro-program can then store it into the old PSW.
24. The left-most byte of the current PSW is put back into Auxiliary Storage with a Store P/ operation.
25. The 2-bit in DR-P is now set to zero because the first part of the interrupt operation is terminated.
26. Control 15 sets the previously generated address /091/ (step 11) into DR-S, T, R.
3. The micro-program enters its loop again at step 3 and fetches byte /091/ out of the Main Storage to clear this position.
4. A Control 15 operation (not shown) brings back half byte Y6 and Y5 (DA and FS) and Y6 and Y5 are set into DR-U and L.
5. A Store STR operation stores the Y6 and Y5 half bytes into the old PSW using address /091/.
6. Only 1 byte of the current PSW is left to be stored into the old PSW. A R-1 micro-routine modifies the address to /090/.
7. Position /090/ of the Main Storage is cleared by a Fetch STR operation.
8. Y8 and Y7, which were located in DR-E, S are moved into DR-U and L.
9. The content of DR-U and L is stored into the old PSW by using address /090/. The complete current PSW is now contained in the old PSW, in fact it is now the old PSW.
10. The test of the 2-bit in DR-P reveals that there is no 2-bit since it was set to zero in Step 25.
33. The micro-program proceeds by generating the address of the new PSW left-most byte. The new PSW will become the current PSW.
34. The Utility byte from Auxiliary Storage position /04/ is fetched and a 1-bit is set into it to indicate that the following operations are not identical with a Set

PSW operation because it shares a common loop with the Exchange PSW operation.

35. A 2-bit is set into DR-P to enable the micro-program to repeat (or not repeat) a specific series of operations.
36. With a Fetch and an immediately following Store STR micro-instruction, the X8 and X7 half-bytes are set into DR-U and L without destroying the new PSW in the Main Storage.
37. The content of DR-U and L is set into DR-E and S. The half-bytes X8 and X7 are now in DR-E, S.
38. An address exchange (Control 15) sets the address /094/ back into DR-S, T, R. This is not shown. Address /094/ is modified by STR+1 Sub-routine to /095/.
39. Using address /095/, a Fetch and Store operation sets the X6 and X5 half-bytes into DR-U, L without destroying the content of the new PSW in Main Storage.
40. The X6 and X5 half-bytes are moved from DR-U, L into DR-T and R. With that, the first half of the new PSW is located in DR-E, S, T, R.
41. A test checks DR-P for a 1-bit. Since DR-P contains only a 2-bit, the micro-program performs the ESTR to Reg 2 Sub-routine. This routine places the content of DR-E, S, T, R (Half of the new PSW) into the first half of the current PSW position in Auxiliary Storage. When this is completed, half of the current PSW is updated with the new PSW.
43. The address /095/ is increased to/ 096/ in preparation for a repetition of the last operation.
44. A 3 is set into DR-P, that is a 1-bit and a 2-bit. This will allow the micro-program to leave the loop.
36. The micro-program now enters its loop again at step 36. With address/096/, the X4 and X3 half-bytes are fetched from the new PSW in Main Storage.
37. The content of DR-U and L is moved to DR-E and S as before. With that, half of the NSI address of the new PSW is located in DR-E and S.
38. The address is increased once more to /097/.
39. The second half of the NSI of the new PSW is read out of Main Storage and immediately regenerated by storing it back.
40. The content of DR-U and L (X2 and Xl) is set into DR-T, R. The NSI address of the new PSW is now located in DR-E, S, T, R which was the purpose of the interrupt routine. In effect, the interrupt branched to a new NSI.
41. DR-P now contains a 1-bit and the micro-program leaves the loop.
45. The Utility byte is read out into DR-U and L and immediately restored. The content of the Utility byte is needed for a test.
46. A test is performed on the content of DR-U and L to see whether or not the last operation was a Set PSW.

    The micro-program recognizes that this was not a Set PSW routine.
47. Finally DR-P is set to zero. The micro-program continues in the I-phase. This completes the Interrupt micro-program.

Upon completion of the Interrupt micro-program, the current PSW contains a channel mask which has no 1-bit. The DR-E, S, T, R contain the address of the next sequential instruction (From the new PSW) to which the machine program branches. In this branch, various tests are performed on the I/O device which caused the interrupt. It is possible that meanwhile another I/O device has reached a channel end condition. This channel end condition, however, cannot cause an interrupt because there is no 1-bit in the channel mask.

This means that the Interrupt is intentionally disabled. If the Interrupt were not disabled an Interrupt within the Interrupt could possibly occur.

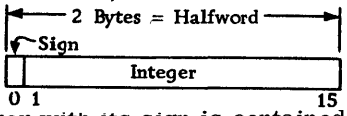
When the branch program is completed, a Set PSW instruction replaces the current PSW with the old PSW. The NSI address of the old PSW leads the machine program back to the point at which it was interrupted. Now the channel mask has a 1-bit again and the Interrupt is active again. Thus any other I/O device that has meanwhile reached a channel end condition may cause a new Interrupt.

ARITHMETIC PRINCIPLES

- Two types of arithmetic operations are performed by the Model 20 CPU:Binary and Decimal.
- Both types of arithmetic operations are implemented with the "Adder"subroutine micro program.
- Binary Arithmetic is limited to a result no greater than can be represented by 15 binary positions (32, 767 decimal).
- The Binary Arithmetic is generally used for calculating addresses.
- Decimal Arithmetic is limited to a result no greater than 31 decimal positions.

Binary Arithmetic

- All fixed point arithmetic (add, subtract, multiply, divide, and compare) operations are accomplished by a binary Adder micro-program.
- Fixed point numbers (Integers) range up to 15 bits plus sign bit having a decimal value of + 32, 767 or -32, 768.



- An Integer with its sign is contained in a half word and can be located either in a General Register or in Main storage.
- The address of a half word in main storage must be even (an integral boundry of 0).
- The half word at the Operand 2 address, is arithmetically operated upon in conjunction with the half word at the Operand 1 address. (One Operand is always located in General Register). The data at the Operand 2 address remains unchanged. The original data at the Operand 1 address is destroyed and is replaced by the result of the arithmetic operation.
- A zero (no) bit in the sign position indicates that the half word contains a positive (plus) value. A one bit in the sign position indicates that the half word contains a negative (minus) value.
- Negative integers are represented in "twos complement" form.
- A Load Complement machine instruction (Op Code 13) complements an integer (plus to minus and vice versa).

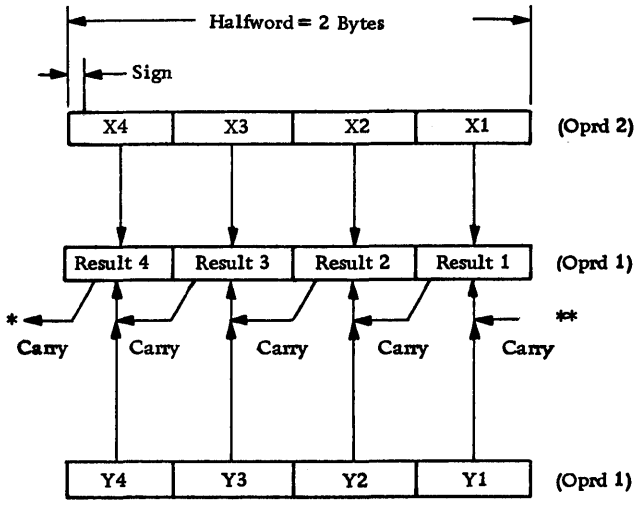
Description (Figures 3-39a and 3-39b)

A plus integer is changed into a minus integer (represented in twos complement) and vice versa by complementing the integer and its sign. Complementing is accomplished by adding a 1 bit into the right most position of the Integer. When a carry occurs it is propogated as necessary up to the highest order position.

Example: Convert a plus 2520 into a minus 2520 and back into a plus 2520 again (Figure 3-39b).

Common Description for Binary Arithmetic

Addition or subtraction of half a word (16 bits) is accomplished by separating the bits into four groups of 4 bits and then performing the arithmetical operations one group at a time because only 4 bits can be processed at one time in the Adder subroutine micro-program. The carry of one group to the other is preserved in the Adder subroutine micro program and is processed during the addition of the next group. In all E phase flow charts, groups of one Operand are labeled X4, X3, X2 and X1; the groups of the other Operand are labeled Y4, Y3, Y2, and Y1. The result of adding or subtracting one X group to one Y group, i.e., Y1 plus X1 is called Result 1. The labeling X, Y, and Result is used to simplify the flow charts of the machine program E phases.



*This carry is ignored.
**A carry into Result 1 (carry = a 1 bit) is always inserted in a subtract operation to generate the twos complement.

Decimal	Binary																
		Halfword															
		Integer															
	Sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
0 (Always Plus)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
+1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
+2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
-2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
+11	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
-11	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1
+1000	0	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0
-1000	1	1	1	1	1	1	0	0	0	0	0	1	1	0	0	0
+32767 } Highest Value of a	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-32768 } 15 Bit Integer	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 3-39a Decimal - Binary Table

	Sign	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
+2520	0	0	0	0	1	0	0	1	1	1	0	1	1	0	0	0
Invert	1	1	1	1	0	1	1	0	0	0	1	0	0	1	1	1
+1																1
Twos Compl. = - 2520	1	1	1	1	0	1	1	0	0	0	1	0	1	0	0	0
Complement	0	0	0	0	1	0	0	1	1	1	0	1	0	1	1	1
+1																1
= +2520	0	0	0	0	1	0	0	1	1	1	0	1	1	0	0	0

Figure 3 - 39b Complementing an Integer

Binary Addition

Only 4 bit integers are shown in all of the following examples.

Adding 1 or 0 to a 0. If there is a "one" (1) in any position (sign, 8, 4, 2, 1) in Operand 1 or Operand 2 and there is a "zero" (0) in the corresponding position of the other operand the result (sum) is "one". If "zeros" (0) are in corresponding operand position the result is zero (0).

	S 8 4 2 1	
+2	0 0 0 1 0	Operand 2 Unchanged
+ +1	0 0 0 0 1	Operand 1 Before
= +3	0 0 0 1 1	Operand 1 After

Adding a 1 to a 1. If corresponding positions of Operand 1 and Operand 2 contain "ones" (1) the result (sum) is zero with a carry. A carry is propogated into the next high order position as a 1 bit.

	S 8 4 2 1	
+2	0 0 0 1 0	Operand 2 Unchanged
+ +3	0 0 0 1 1	Operand 1 Before
= +5	0 0 1 0 1	Operand 1 After

Adding a carry into a position which contains corresponding 1 bit. A carry is propogated as a binary 1 bit. The sum of three "one" bits gives a "one" bit and a carry.

	S 8 4 2 1	
+3	0 0 0 1 1	Operand 2 Unchanged
+ +7	0 0 1 1 1	Operand 1 Before
= +10	0 1 0 1 0	Operand 1 After

It is not a proper operation to add two (plus) integers together, if the result is greater than the machine can represent in 15 bits (+32,767). In the examples used here (4 bits) the result is +15.

If the correct result of an addition is greater than +32,767 both the sign and the sum of the result will be incorrect. The sign of the result will be indicated as 1 minus when both operand signs are plus. A test is made of the three signs (Operand 1, Operand 2, and result) to see if an improper addition has taken place. See Op code /1A/ and /4A/. If an improper addition has taken place an /8/ is moved into DR-I to indicate "Binary Overflow." The Binary Overflow condition stops the CPU.

			S 8 4 2 1	
+9		+9	0 1 0 0 1	Operand 2 Unchanged
++ +8	+ +	+8	0 1 0 0 0	Operand 1 Before
= +17	=	-15	1 0 0 0 1	Operand 1 After
Correct Result	Incorrect Result		This bit configuration represents a -15 in two complement.	

Adding a minus integer to a plus integer having a greater absolute value. The result sign, of adding a minus integer to a plus integer having a greater absolute value is always plus.

A carry from the high order position of an integer is propagated into the sign position, thus changing the result sign.

	S 8 4 2 1	
-3	1 1 1 0 1	Operand 2 Unchanged
+ +6	0 0 1 1 0	Operand 1 Before
= +3	0 0 0 1 1	Operand 1 After

Adding a minus integer to a plus integer having a smaller absolute value. The result is always minus, and is represented in twos complement.

	S 8 4 2 1	
-6	1 1 0 1 0	Operand 2 Unchanged
+3	0 0 0 1 1	Operand 1 Before
= -3	1 1 1 0 1	Operand 1 After

Adding a minus integer to a minus minus. The result is always minus, and is represented in twos complement.

	S 8 4 2 1
-6	1 1 0 1 0
+ -3	1 1 1 0 1
= -9	1 0 1 1 1

It is not a proper operation to add two minus integers together, if the result is greater than the machine can represent in 15 bits (32,768). In the examples used here (4 bits) the result is -16.

If the correct result of an addition is greater than 32,768 both the sign and the sum of the result will be incorrect. The sign of the result will be indicated as plus when both operand signs are minus. A test is made of the three signs (Operand 1, Operand 2, and result) to see if an improper addition has taken place. See Op code /1A/ and /1B/. If an improper addition has taken place, an /8/ is moved into DR-I to indicate "Binary Overflow." The Binary Overflow condition stops the CPU.

			S 8 4 2 1	
-9	-9		1 0 1 1 1	Operand 2 Unchanged
+ -8	+ -8		1 1 0 0 0	Operand Before
= -17	= +14		0 1 1 1 1	Operand After
Correct Result	Incorrect Result			

Binary Subtraction

A binary Integer is subtracted as shown in the accompanying example (Integer length is 4 bit only).

NOTE: A subtract in binary arithmetic is a normal addition of the Operand 1 bits and Operand 2 bits. Operand 2 bits are converted into twos complement before they are added to the Operand 1 bits.

Subtracting a plus integer from a plus integer having a greater absolute value. The result sign, of subtracting a plus integer from a plus integer having a greater absolute value, is always plus.

Operand 2 in twos complement is added to Operand 1.

			S 8 4 2 1	
+7	+7		0 0 1 1 1	Operand 1 Before
- +4			0 0 1 0 0	Operand 2 Unchanged
= +3		-4	1 1 1 0 0	Operand 2 In Two Complement
	=	+3	0 0 0 1 1	Operand 1 After

Subtracting a plus integer from a plus integer having a smaller absolute value. The result sign, of subtracting a plus integer from a plus integer having a smaller absolute value, is always minus and is represented in twos complement.

A carry from the high order position of an integer is propagated into the sign position, thus changing the result sign. Operand 2 in twos complement is added to Operand 1.

			S 8 4 2 1	
+4	+4		0 0 1 0 0	Operand 1 Before
+7			0 0 1 1 1	Operand 2 Unchanged
= +3	+	-7	1 1 0 0 1	Operand 2 In Twos Complement
	=	-3	1 1 1 0 1	Operand 1 After

It is not a proper operation to subtract a plus integer from a minus integer if the result is greater than the machine can represent in 15 bits (-32,768). In the examples used here (4 bits) the greatest result is -16.

If the correct result of a subtraction is greater than -32,768 both the sign and the sum of the result will be incorrect. The sign of the result will be indicated as plus when one operand sign is minus and the other is plus. A test is made of the three signs (Operand 1, Operand 2, and result) to see if an improper subtraction has taken place. The sign of Operand 2 is complemented to use the same test loop as in a binary address. (See Op code /1B/ and /4B/). If an improper addition has taken place, an /8/ is moved into DR-I to indicate a "Binary Overflow." The Binary Overflow condition stops the CPU.

Operand 2 in twos complement is added to Operand 1.

			S 8 4 2 1	
-8	-8		1 1 0 0 0	Operand 1 Before
- +9			0 1 0 0 1	Operand 2 Unchanged
= -17		+9	1 0 1 1 1	Operand 2 In Twos Complement
	=	+2	0 1 1 1 1	Operand 1 After
Correct Result	Incorrect Result (Binary Overflow)			

Decimal Arithmetic

- All arithmetic operations programmed in SS Format (add, subtract, multiply, divide, and compare) handle data in packed format.
- Unpacked data in boundaries of 16 Bytes can be changed from unpacked to packed format. Packed data is coded in binary form.
- Thirty-two numeric characters (4 Bits each) can be handled by decimal arithmetic operations, however the low-order character is reserved for a sign.
- The data is located in Main Storage and in the Accumulator located in Auxiliary Storage.
- The processor always processes one character of the first operand field and one character of the Accumulator.
- The data in the second operand field remains unchanged.
- The original data in the first operand field is destroyed and is replaced by the result of the arithmetic operation.
- A minus sign is indicated by having 11 or 13 in the sign position.
- A plus sign is indicated by having a 10, 12, 14, or 15 in the sign position.
- The sign can be changed by increasing or decreasing.

Common Description for Decimal Arithmetic

Addition or subtraction of a decimal numeric character (4 bits) is performed by micro-program control (Binary Adder Subroutine). The Binary Adder Subroutine handles data in binary format only. A binary value of 15 initiates a carry. To obtain a decimal value, a six is added if the result is greater than 9 or a carry occurred. To carry, a one is added to the next higher order, decimal position.

```
      +3        0011
      +3        0011
      +6        0110    Correct decimal result.

      +9        1001
      +9        1001
Carry +8        0010    Carry, which initiates adding 6
                0110    Add 6
          Carry 1000

      +6        0110
      +6        0110
Carry +2        1100    Result greater than 9 initiates
                        adding 6
                0110    Add 6
          Carry 0010
```

In all E-phases, the first Operand flow chart groups are labeled $(Y1)_n$ and the sign is labeled $(Y2)_o$. The second Operand groups are labeled $(X1)_n$ and the sign is labeled $(X2)_o$. The result of adding or subtracting one X group to one Y group is called Result n. The X, Y, and Result labeling is used to simplify the flow charts of the machine program E-phases.

Binary Addition (True Add)

Only half bytes are shown in the following examples:

Adding a Bit or No Bit (Operand 1) to No Bit (Operand 2). If there is a one (1) in any bit position (8, 4, 2, 1) in Operand 1 or Operand 2 and a zero (0) in the corresponding position of the other operand, the result (sum) is one. If zeros (0) are in corresponding operand positions, the result is zero (0).

```
                8421
      +2        0010    Operand 2 Unchanged
      +1        0001    Operand 1 Before
  =   +3        0011    Operand 1 After
```

Adding a Bit (Operand 1) to a Bit (Operand 2). If corresponding positions of Operand 1 and Operand 2 contain bits (1), the result (sum) is zero with a carry. A carry is added into the next higher order position as a 1-bit.

```
                8421
      +2        0010    Operand 2 Unchanged
      +3        0011    Operand 1 Before
  =   +5        0101    Operand 1 After
```

Adding a Carry Into a Position Which Contains a 1-Bit. A carry is added as a binary 1. The sum of three bits gives a bit and a carry.

```
                8421
      +3        0011    Operand 2 Unchanged
     ++7        0111    Operand 1 Before
      +10       1010    Operand 1 After
```

The operand 1 and 2 can not exceed 16 bytes corresponding to 31 decimal positions plus sign. If the result exceeds the field length of Operand 1, the condition code is set to 11 indicating an overflow.

The Operand 2 field sign is the resulting sign.

Binary Subtraction (Complement Add)

A binary half byte is subtracted as shown in the accompanying example.

A subtract in decimal arithmetic is a normal addition of the Operand 1 bits and Operand 2 bits. However, Operand 2 bits are converted into nine's complement form before they are added to the Operand 1 bits.

Complement Add with Operand 1 smaller than Operand 2. No carry occurs and the result is not in true form. In order to get the right decimal value, a Recomplement is accomplished and the sign of Operand 2 is changed and stored as the sign of the result.

```
                8421
      +4        0100    Operand 1 Before
  -   +7        0111    Operand 2 Unchanged
  =   -3        0010    Operand 2 In nine's
                        Complement Form
                0110    No Carry Start Recomplement
                0011
```

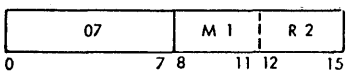
Complement Add with Operand 1 Greater than Operand 2. A carry occurs and the result has the right value. The sign of Operand 2 is stored as the sign of the result.

```
                8421
      +7        0111    Operand 1 Before
  -   +4        0100    Operand 2 Unchanged
  =   +3        0101    Operand 2 In nine's
                        Complement Form
                1100    Result greater than 9, add 6
                0110
          Carry 0010
                0011    Carry
```

E-PHASE OPERATIONS

(Brief descriptions of objectives of macro-program Op codes will be included when available.)

Branch On Condition (BCR) Branch RR Format



- Objective: The updated instruction address is replaced by the branch address if the state of the condition code is as specified by M1; otherwise, normal instruction sequencing proceeds with the updated instruction address. (00000111)
- The Op code is 07.
- M1 is a four bit field, used as a Mask.
- R2 is the address of a GR which contains the Branch address.

Description (SLD 20-27)

The M1 field is used as a four-bit mask. The four bits of the mask correspond, left to right, with the four condition codes shown in the following:

Condition Code	Instruction Bits
0 0	8
0 1	9
1 0	10
1 1	11

The branch is successful (occurs) whenever the condition code has a corresponding mask bit of one.

Example: To cause a branch to occur due to a zero result of an add operation M1 = 1000.

Example: To cause a branch to occur due to a zero result or due to a result greater than zero of an add operation M1 = 1 0 1 0.

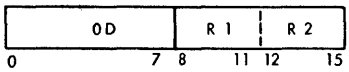
Programming Note

When all four mask bits are ones the branch is unconditional. When all four mask bits are zero or when the R2 field contains zero, the branch instruction is equivalent to a No Operation.

Condition Code

The condition code remains unchanged.

Branch and Store (BASR) Branch RR Format



- Objective: The rightmost sixteen bits of the PSW, the updated instruction address, are stored as link information in the general register specified by R1. Subsequently, the instruction address is replaced by the branch address.
- The Op code is 0D (00001101).
- R1 is the address of a GR into which the NSI address is stored.
- R2 is the address of GR which contains the Branch address.

Description (SLD 20-25)

The branch address is determined prior to the storing of the link information.

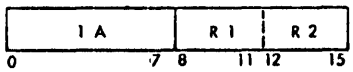
Programming Note

When the R2 field contains zero, the link information is stored without branching.

Condition Code

The condition code remains unchanged.

Add (AR), Fixed Point RR Format

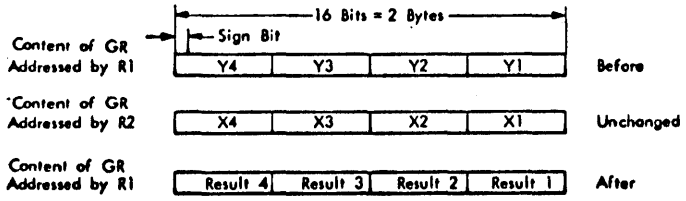


- Objective: Add the second operand to the first operand.
- The sum is placed in the first operand location.
- The Op code is /1A/ (00011010).
- R1 is the address of a GR which contains Operand 1.
- R2 is the address of a GR which contains Operand 2.

Description (SLD 20-29)

Operands and sums are treated as fifteen bit integers with sign. Addition is performed by adding all sixteen bits of both operands. If the carry out of the sign-bit position and the high-order numeric bit agree, the sum is satisfactory. If they disagree an overflow occurred. A positive overflow yields a negative final sum and a negative overflow results in a positive sum.

An overflow results in a Binary Overflow error condition. The CPU stops and DR-I contains an /8/ to indicate the binary overflow.



Programming Note

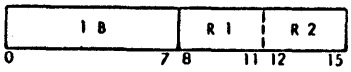
In two's complement notation, a zero result is always positive.

Condition Code

The condition code is set according to the result of the addition:

Result	Condition Code
Sum is zero	0 0
Sum is less than zero	0 1
Sum is greater than zero	1 0

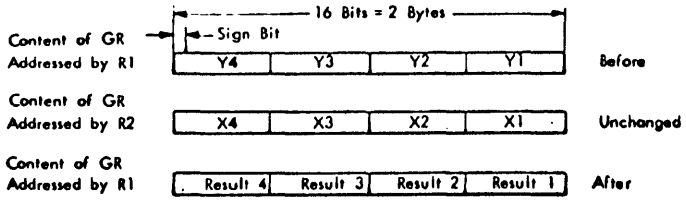
Subtract (SR), Fixed Point RR Format



- Objective: Subtract the second operand from the first operand.
- The difference is placed in the first operand location.
- The Op code is /1B/ (00011011).
- R1 is the address of a GR which contains Operand 1.
- R2 is the address of a GR which contains Operand 2.

Description (SLD 20-28)

Operands and differences are treated as fifteen bit integers with sign. Subtraction is performed by adding the two's complement of the second operand to the first operand. All sixteen bits of both operands participate as in the Add instruction. If the carry out of the sign-bit position and the high-order numeric bit position agree, the difference is satisfactory. If they disagree, an overflow occurred, resulting in a Binary Overflow error condition. The Binary Overflow condition stops the CPU and DR-I contains an /8/ to indicate a binary overflow.



Programming Note

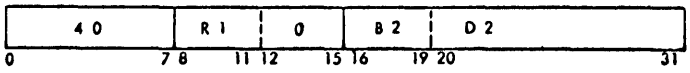
When the same register is specified as first and second operand location, subtracting is equivalent clearing the register. Subtracting a maximum negative number from another maximum negative number gives a zero result and no overflow.

Condition Code

The condition code is set according to the result of the subtraction:

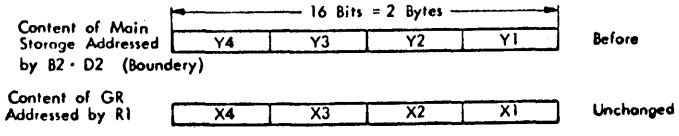
Result	Condition Code
Difference is zero	0 0
Difference is less than zero	0 1
Difference is greater than zero	1 0

Store Half word (STH), Fixed Point RX Format



- Objective: Store the first operand at the halfword second operand location.
- The Op code is /40/ (0100 0000).
- R1 is the address of a GR which contains Operand 1.
- The four-bit field is not used and must be /0/ (0000).
- B2 and D2 is the direct or effective Main-storage address of Operand 2.

Description (SLD 20-25)



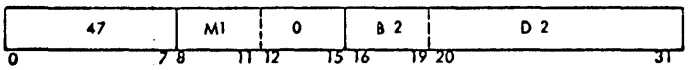
Programming Note.

In this operation the Operand 2 and not Operand 1 is replaced (destroyed). If the four bit field (12-15) is not /0/, the CPU stops with a /6/ in DR-I to indicate a program (specification) error.

Condition Code

The condition code remains unchanged.

Branch on Condition (BC), Branch RX Format



- The updated instruction address (NSI address) is replaced by the branch address if the state of the condition code is as specified by M1; otherwise, normal instruction sequencing proceeds with the NSI address.
- The Op code is /47/ (0100 0111).
- M1 is a four bit field used as a mask.
- The /0/ four bit field is not used and must be /0/ (0000).
- B2 and D2 is a direct or effective main storage address which is used as a branch address.

Description (SLD 20-13)

The M1 field is used as a four-bit mask. The four bits of the mask correspond, left to right, with the four condition codes, 0, 1, 2, and 3, as shown in the following table.

Condition Code	Instruction Bits
0 0	8
0 1	9
1 0	10
1 1	11

The branch is successful whever the condition code has a corresponding mask bit of one.

Example: To branch on equal result of a compare, M1 = 1000.

Example: To branch on equal or first operand low, M1 = 1100.

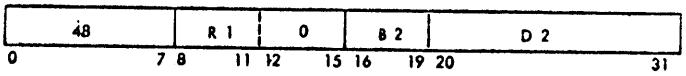
Programming Note

When all four mask bits are ones, the branch is unconditional. When all four mask bits are zero, the branch instruction is equivalent to a no operation.

Condition Code

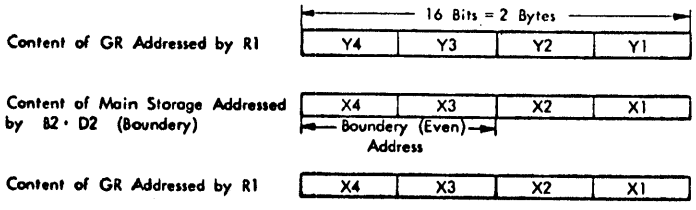
The condition code is not changed.

Load Halfword (LH) Fixed Point RX Format



- The halfword second operand is placed in the first operand location.
- The Op code is /48/ (0100 1000).
- R1 is the address of a GR which contains the Operand 1.
- The /0/ four bit field is not used and must be /0/ (0000).
- B2 and D2 is the direct or effective main storage address of Operand 2.

Description (SLD 20-13)



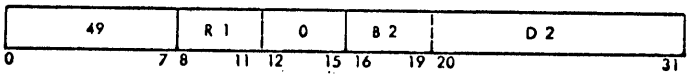
Programming Notes

1. If the four bit field (12-15) is not /0/, the CPU stops with a /6/ in DR-I to indicate a program (specification) error.
2. The Operand 2 address must be even (boundary). If it is not even the CPU stops. DR-I contains a /6/ to indicate a program (specification) error.

Condition Code

The condition code remains unchanged.

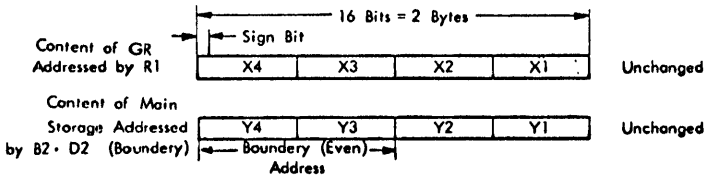
Compare Halfword (CH), Fixed Point RX Format



- The first operand is compared with the second operand.
- The result determines the setting of the condition code.
- The Op code is /49/ (0100 1001).
- R1 is the address of a GR which contains Operand 1.
- The /0/ four bit field is not used and must be /0/ (0000).
- B2 and D2 is a direct or effective main-storage address of Operand 2.

Description (SLD 20-22)

Comparison is algebraic. Both operands are treated as fifteen bit integers with signs. Operands in registers or storage are not changed as a result of the operation.



Programming Note

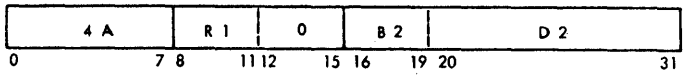
If the four bit field (12-15) is not /0/, the CPU stops with a /6/ in DR-I to indicate a program (specification) error.

Condition Code

The condition code is set according to the result of the operation:

Result	Condition Code
Equal	0 0
First Operand is low	0 1
First Operand is high	1 0

Add Halfword (AH), Fixed Point RX Format

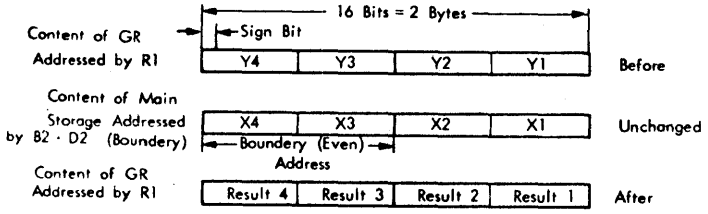


- The second operand is added to the first operand.
- The sum is placed in the first operand location.
- The Op code is /4A/ (0100 1010).
- R1 is the address of a GR which contains Operand 1.
- The /0/ four bit field is not used and must be /0/ (0000).
- B2 and D2 is a direct or effective Main-storage address and must be even (boundary).

Description (SLD 20-21)

Operands and sums are treated as fifteen bit integers with signs. Addition is performed by adding all sixteen bits of both operands. If the carries out of the sign-bit position and the high-order numeric bit position agree, the sum is satisfactory. If they disagree, an overflow occurred. The sign bit is not changed after the overflow. A positive overflow yields a negative final sum and a negative overflow results in a positive sum. An overflow results in a Binary Overflow error condition.

The Binary Overflow condition stops the CPU and DR-I contains an /8/ to indicate Binary Overflow.



Programming Note

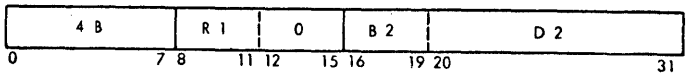
In two's complement notation, a zero result is always positive.

Condition Code

The condition code is set according to the result of the addition:

Result	Condition Code
Sum is zero	0 0
Sum is less than zero	0 1
Sum is greater than zero	1 0

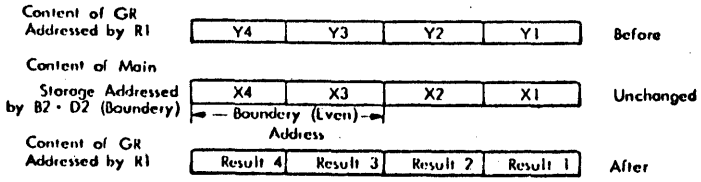
Subtract Halfword (SH), Fixed Point RX Format



- The second operand is subcontracted from the first operand, and the difference is placed in the first operand location.
- The Op code is/4B/(0100 1011).
- R1 is the address of a GR which contains Operand 1.
- The /0/ four bit field is not used and must be /0/ (0000).
- B2 and D2 is a direct or effective main storage address of Operand 1 and must be even (boundary).

Description (SLD 20-20)

Operands and differences are treated as fifteen bit integers with a sign. Subtraction is performed by adding the two's complement of the second operand to the first operand. All sixteen bits of both operands participate as in the instruction add. If the carries out of the sign-bit position and the high-order numeric bit position agree, the difference is satisfactory. If they disagree, an overflow occurred, resulting in a Binary Overflow error condition. The Binary Overflow condition stops the CPU and DR-I contains an /8/ to indicate Binary Overflow.



Programming Note

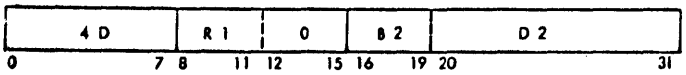
Subcontracting a maximum negative numbers from another maximum negative number gives a zero result and no overflow.

Condition Code

The condition code is set according to the result of the subtraction:

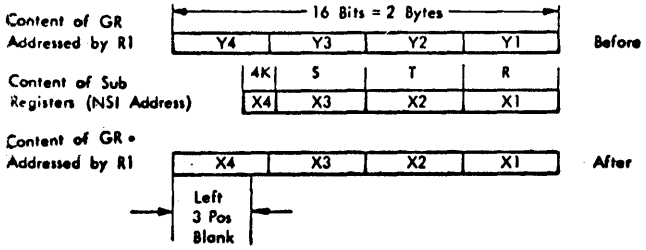
Result	Condition Code
Difference is zero	0 0
Difference is less than zero	0 1
Difference is greater than zero	1 0

Branch and Store (BAS), Branch RX Format



- The rightmost sixteen bits of the PSW, the updated instruction address, are stored as link information in the general register specified by R1.
- Subsequently, the instruction address is replaced by the branch address.
- The Op code is /4D/ (0100 1101).
- R1 is the address of a GR which receives the NSI address.
- The four bit field is not used and must be /0/ (0000).
- B2 and D2 is a direct or effective main storage address which is used as branch address.

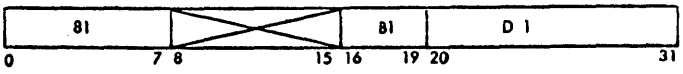
Description (SLD 20-27)



Condition Code

The condition code remains unchanged.

Set PSW (SPSW), Branch SI Format



- The 32-bit word (four eight-bit bytes) located in main storage with the leftmost byte at the first operand address replaces the program status word (PSW).
- The Op code is /81/ (1000 0001).
- Bits 8 through 15 are ignored.
- B1 and D1 is the direct or effective main-storage address of leftmost byte of a located PSW.

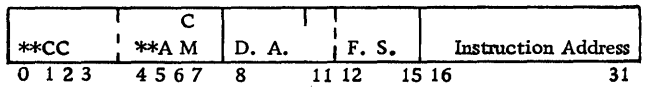
Description (SLD 20-19)

The Set PSW instruction is equivalent to a branch operation.

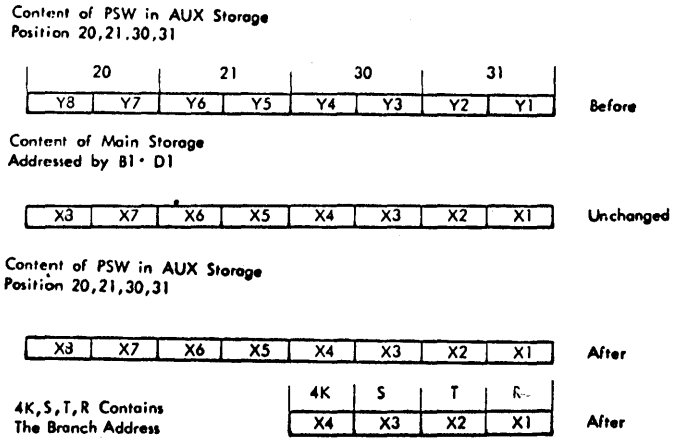
When an input/output interrupt occurs, the PSW is stored in Main Storage locations 144-147 and a new PSW is obtained from Main Storage locations 148-151.

The PSW has a fixed-length format of one word. It is located in an unaddressable (by machine program) register in the CPU (AUX Storage position 20, 21, 30, 31) and is employed as an internal control.

PSW Format



0 - 1	Not used
2 - 3	Condition Code
4 - 5	Not used
6	ASCII Mode Bit
7	Channel Mask
8 - 11	Device Address (Input/Output Interrupt)
12 - 15	Function Specification (Input/Output Interrupt)
16 - 31	Instruction Address



Programming Note

The instruction address portion of the word which is transferred from main storage to the PSW by the Set PSW operation should:

1. Not refer to the protected (first 144) bytes of main storage.
2. Have the least-significant bit zero.
3. Be within the limits of available storage.

If these conditions are not satisfied, program (address or specification) error stop will occur.

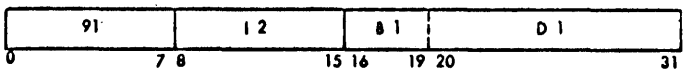
Main Storage boundaries are not required of the first operand address in the set PSW instruction.

The condition code, ASCII Mode Bit, and Channel Mask in the PSW are zero when the CPU is in the reset state. The instruction address portion of the PSW is not changed when the CPU is reset.

Condition Code

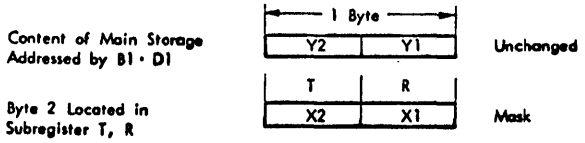
The condition code is set by the Set PSW operation, to the value contained in the word transferred from Main Storage to the PSW.

Test under Mask (TM) Logical Data, SI Format



- The first operand one byte is ANDed with the second operand (one byte) to set the condition code.
- No operand is destroyed.
- The Op code is /91/ (1001 0001).
- I2 is the Operand 2 and is called Mask.
- B1 and D1 is a direct or effective main-storage address of Operand 1.

Description (SLD 20-18)



Condition Code

The condition code is set to:

00 = Zero, if Operand 1 and Operand 2 (Mask) have no corresponding bits.

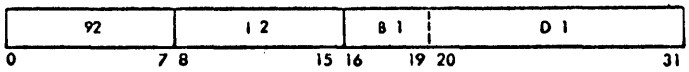
11 = All ones, if Operand 1 has bits (1) in all corresponding positions where the Operand 2 (Mask) has bits (1).

01 = Mixed, for all other bit patterns in Operand 1 or Operand 2.

Example for All Ones

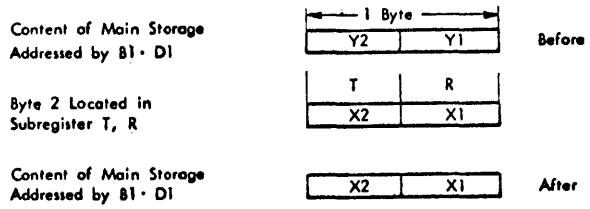
Operand 1	10101011
Operand 2 (Mask)	00100010

Move (MVI) Logical Data, SI Format



- The second operand is placed in the first operand location.
- The Op code is /92/ (1001 0010).
- I2 is the Operand 2.
- B1 and D1 is the direct or effective main storage address of Operand 1.

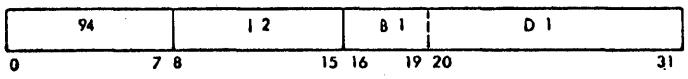
Description (SLD 20-12)



Condition Code

The condition code remains unchanged.

AND (NI) Logical Data, SI Format



- The logical product (AND) of the first and second operand bits is placed in the first operand location.
- The Op code is /94/ (1001 0100).
- I2 is an eight bit pattern which is ANDed with the byte addressed by the Operand 1 address.
- B1 and D1 is the direct or effective main storage address of Operand 1.

Description (SLD 20-17)

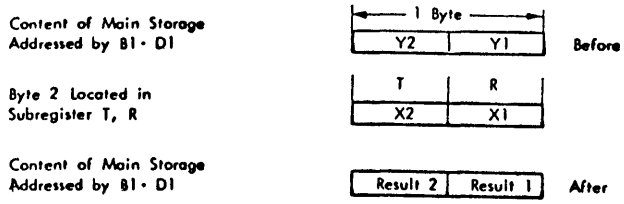
Two examples are given. Example 1 sets the CC to 01 (Not zero), example 2 sets the CC to 00 (zero).

Example 1:

Operand 1	10101010	Before
I2 (Byte 2)	01101101	Unchanged
Operand 2	00101000	After (Result
	Result Not Zero	CC = 01

Example 2:

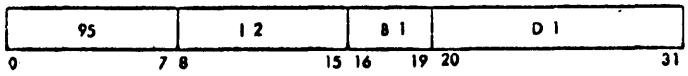
Operand 1	10101010	Before
I2 (Byte 2)	01000101	Unchanged
Operand 2	00000000	After (Result)
	Result zero	CC = 00



Condition Code

The condition code (CC) is set to zero, 00, or not-zero, 01, according to the result of the operation.

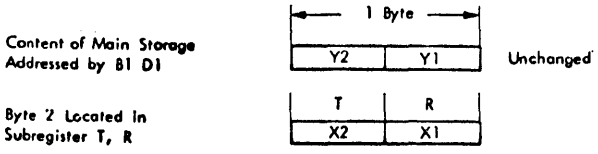
Compare (CLI) Logical Data, SI Format



- The first operand is compared with the second operand.
- The result is indicated in the condition code.
- The Op code is /95/ (1001 0101).
- I2 is the Operand 2.
- B1 and D1 is the direct or effective main storage address of Operand 1.

Description (SLD 20-16)

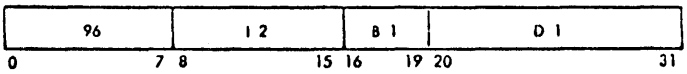
The comparison is made with both operands in binary form.



Condition Code

The condition code is made 00 if the operands are equal, 01 if the first operand is low compared to the second operand, and 10 if the first operand is high compared to the second operand.

OR (OI) Logical Data, SI Format



- The logical sum (or) of the first and second operand bits is placed in the first operand location.
- The Op code is /96/ (1001 0110).
- I2 is an eight bit pattern which is ORed with the byte addressed by the Operand 1 address.
- B1 and D1 is the direct or effective Main s Storage address of Operand 1.

Description (SLD 20-15)

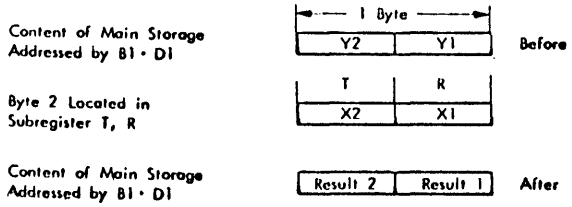
Two examples are given. Example 1 sets the CC to 01 (Not zero). Example 2 sets the CC to 00 (zero).

Example 1:

Operand 1	10101010	Before
I2 (Byte 2)	01101101	Unchanged
Operand 1	11101111	After (Result)
	Result Not Zero	CC = 01

Example 2:

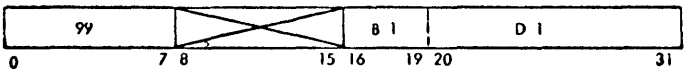
Operand 1	00000000	Before
I2 (Byte 2)	00000000	Unchanged
Operand 1	00000000	After (Result)
	Result Zero	CC = 00



Condition Code

The condition code is set to zero (00), or not-zero (01) according to the result of the operation.

Halt and Proceed (HPR) Logical SI Format



- Stop the CPU for customer control purposes.
- The Op code is /99/ (1001 1001).
- Bits 8 through 15 are ignored.

- B1 and D1 is the direct or effective address and is located in DR-A, S, T, R when the CPU stops.

Description (SLD 20-12)

Condition Code

The condition code remains unchanged.

Move Numerics (MVN) Logical SS-Format

(See A Next Page)

- Objective: The low-order four bits of each byte in the second operand field, the numerics, are placed in the low-order bit positions of the bytes in the first operand field. The high-order four bits of each byte, the zones, remain unchanged in both operand fields.
- D1 (1101 0001) is the Op code.
- L is the field length of both Operands.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-31)

The instruction has the SS format, and therefore is a storage-to-storage move. Movement is left to right through each field. The fields may overlap in any desired way.

(See B Next Page)

Condition Code

The condition code is not changed. The numerics are not changed or checked for validity.

Move (MVC) Logical SS-Format

(See C Next Page)

- Objective: The second operand is placed in the first operand location.
- The SS format is used for storage-to-storage move.
- In the storage-to-storage move, the fields may overlap in any desired way. Movement is left to right through each field a byte at a time.
- The bytes to be moved are not changed or inspected. The condition code is not changed.
- D2 (1101 0010) is the Op code.
- L is the field length of both Operands.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-31)

(See D Next Page)

Programming Note

It is possible to propagate one character through an entire field by having the first operand field start one character to the right of the second operand field.

Condition Code

The condition code remains unchanged.

Move Zones (MVZ) Logical SS-Format

(See E Next Page)

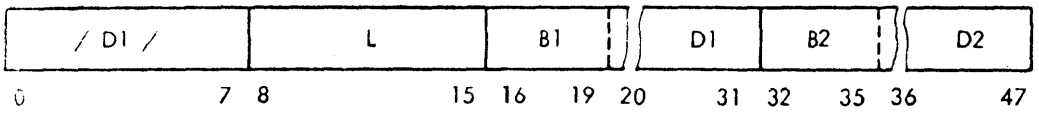
- Objective: The high-order four bits of each byte in the second operand field, the zones, are placed in the corresponding bit positions of the first operand field. The low-order four bits of each byte, the numerics, remain unchanged in both operand fields.
- The Op Code is D3 (1101 0011).
- L is the field length of both Operands.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-32)

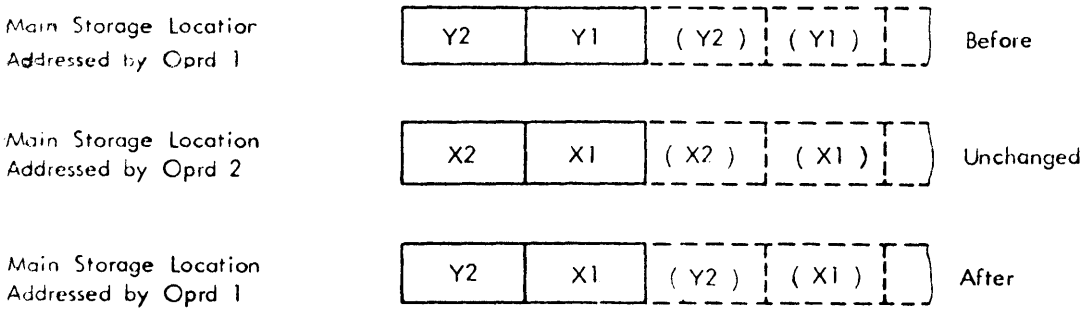
The instruction has the SS format and therefore is storage-to-storage. Movement is left to right through each field and the same overlapping field conditions may arise as in the preceding move instruction.

(See F Next Page)

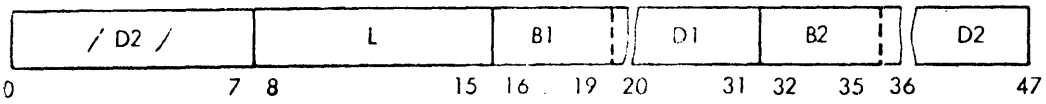
NOTE: All illustrations for this page are on the following page.



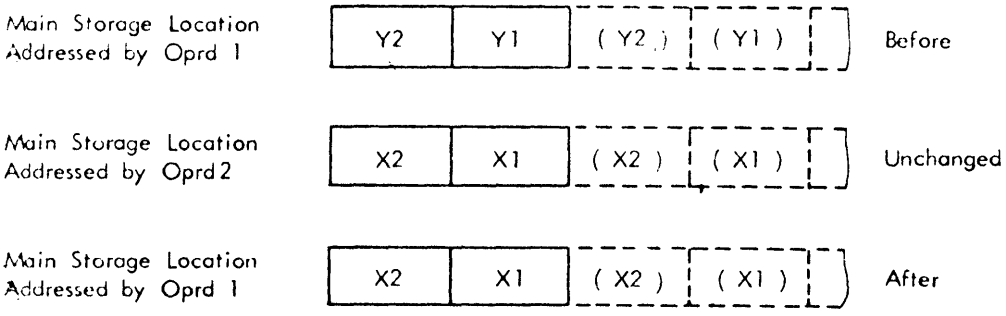
A



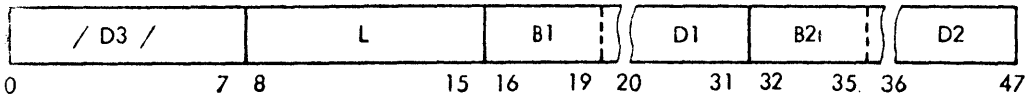
B



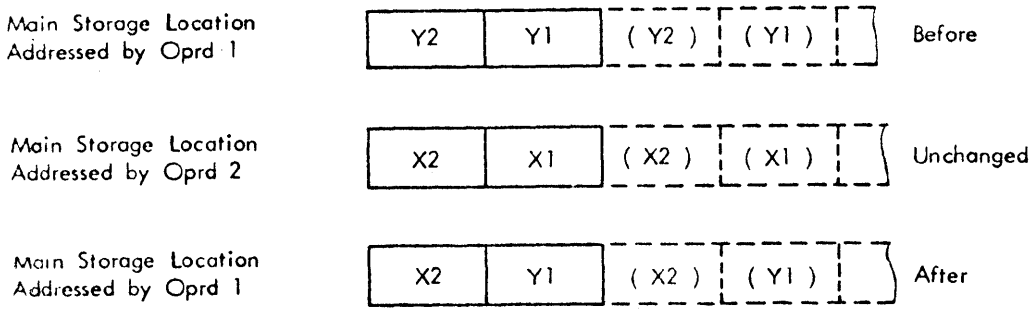
C



D



E



F

Condition Code

The condition bits are not changed.

Validity Checking

The zones are not changed or checked for validity.

Compare Logical (CLC) Logical SS-Format

(See A Next Page)

- Objective: The first operand is compared with the second operand, and the result is indicated in the condition code.
- The Op code is D5 (1101 0101).
- L is the field length of both Operands.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-33)

The SS format is used for storage-to-storage comparison. The operation proceeds left to right and terminates as soon as an inequality is found.

(See B Next Page)

Programming Note

In the Compare Logical operation, all bits are treated alike as part of an unsigned binary quantity. In the variable length storage-to-storage operation, comparison is left to right and may extend to field lengths of 256 bytes. The operation may be used for alphanumeric comparison.

Condition Code

The condition code is made 00 if the operands are equal, 01 if the first operand is low compared to the second operand, and 10 if the first operand compares high.

Validity Checking

Comparison is binary and treats all codes as valid.

Translate (TR), Logical, SS Format

(See C Next Page)

- Objective: The second operand address designates the beginning of a Translate List. The binary value of each byte of the first operand selects a position within this list. The contents of this position replaces the selecting byte in the first operand.
- The Op code is DC (1101 1100).
- L is the field length of both Operands.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

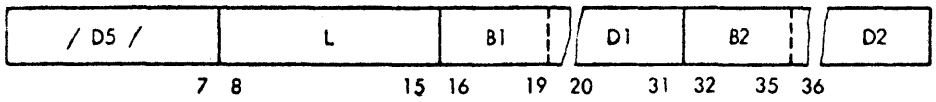
Description (SLD 20-38)

The bytes of the first operand are selected one by one for translation. Each argument byte is added to the entire initial address, the second operand address, in the low-order bit positions. The sum is used as the address of the function byte which then replaces the original argument byte. The operation proceeds until the first operand field is exhausted. It is permissable for the list and the first operand field to overlap.

(See D Next Page)

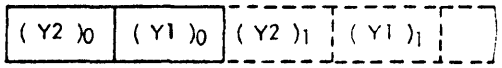
NOTE: All the illustrations for this page are on the following page.

A



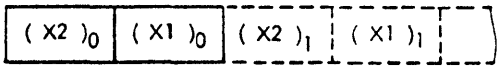
B

Main Storage Location Addressed by Oprd 1



Unchanged

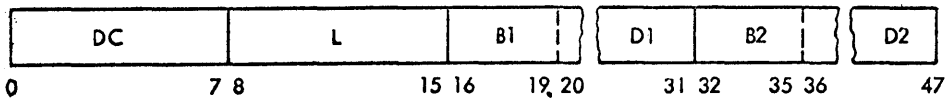
Main Storage Location Addressed by Oprd 2



Unchanged

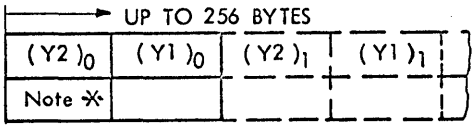
Condition Code is set by the result.

C



D

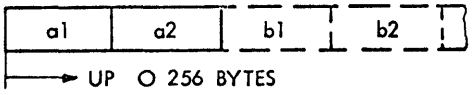
Argument (In Main Storage Location Addressed by Oprd 1 Addr)



Before

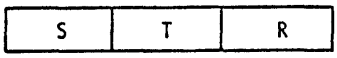
After

Table



Unchanged

Table Address (Oprd 2 Addr)



Argument +

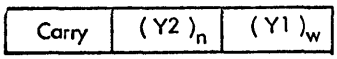
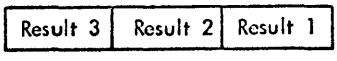


Table Position Addressed by



⋇Contents of Oprd 1 Depends on the Byte in the Table Addressed by Result

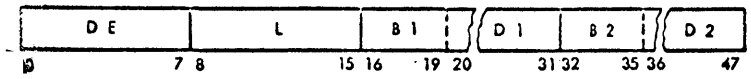
Programming Note

In most applications, the contents of the translation list are established in advance, and the content of the first operand field are unknown data. By interchanging these roles, a reordering rather than a translation is obtained. In that case the data are addressed by the second operand address and thus treated as the contents of a list. The first operand address refers to a field containing control bytes. These control bytes in effect specify the order in which the data bytes are to be placed.

Validity Checking. All data are considered valid.

Condition Code. The condition code remains unchanged.

Edit (ED), Logical SS Format



- The format of the source (the second operand) is changed from packed to zoned, and is edited under control of the pattern (the first operand).
- The edited result replaces the pattern.
- Editing includes sign and punctuation control, and the suppressing and protecting of leading zeros.
- Editing facilitates programmed blanking of all-zero fields.
- Several numbers may be edited in one operation, and numeric information may be combined with text.
- The Op code is DE (1101 1111).
- L is the field length of the pattern (First Operand).
- B1 D1 is the main storage address of the pattern.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-57, 20-58)

The length field applies to the pattern (the first operand). The pattern has the unpacked format and may contain any character. The source (the second operand) has the packed format and must contain valid digit and sign codes. The left four bits of a byte must be 0000-1001, otherwise a data error occurs. The right four bits are recognized as either a sign or a digit.

Both operands are processed left to right one character at a time. Overlapping pattern and source fields give unpredictable results.

The character to be stored in the first operand field is determined by three things; the digit obtained from the source field, the pattern character, and the state of a trigger, called the S trigger. One of three actions may be taken:

1. The source digit may be stored.
2. The pattern character may be left unchanged, or
3. A fill character may be stored.

Programming Notes. As a rule, the source operand is shorter than the pattern since it yields two digits or a digit and a sign for each source number.

When a single instruction is used to edit several numbers, the zero-field identification is provided only for the last field.

The following table gives the details of an editing operation. The left-most columns give the pattern character and its code. The next columns show the states of the digit and the S trigger used to determine the resulting action. The right-most column shows the new setting of the S trigger.

Character Code	Name and purpose	Examine Digit	Trigger Status	Digit Status	Result Char.	Trigger Set
0010 0000	Digit Select	yes	s=1		Digit	
			s=0	d not 0	Digit	s=1
			s=0	d=0	Fill	
0010 0001	Significance		s=1		Digit	
	Start	yes	s=0	d not 0	Digit	
			s=0	d=0	Fill	s=1
0010 0010	Field Separator	no			Fill	s=0
Other	Message Insertion	no	s=1 s=0		Leave Fill	

Legend
d - Source digit
s - S trigger 1, minus sign, digits or pattern used, 0, plus sign, fill used
Digit - A source digit replaces the pattern character
Fill - The fill character replaces the pattern character
Leave - The pattern character remains unchanged

S trigger. The S trigger is used to control the storing or replacing of source digit and pattern characters. Source digits are replaced when zero suppression or protection is desired. Digits to be stored in the result, whether zero or not, are termed significant. Pattern characters are replaced or stored when they

are significance-dependent or sign-dependent, such as punctuation or credit symbols. The S trigger also is used to record the sign of the source and set the condition code accordingly.

The S trigger is set to the zero state at the start of the operation, and is subsequently changed depending upon the source number and the pattern characters.

Pattern Character. Three pattern characters have a special use in editing. They are the digit-select character, the significance-start character, and the field-separation character. These three characters are replaced, either by a source digit or by a fill character; their encoding is shown in Table 1.

1. The digit-select character causes either a source digit or the fill character to be inserted in the result field.
2. The significance-start character has the same function. It also indicates that the following digits are significant.
3. The field-separator character identifies individual fields in a multiple-field editing operation. The character is replaced by the fill character. The S trigger is set to zero and testing for a zero-field is reinitiated.
4. All other pattern characters are treated in a common way; if the S trigger is one, the pattern character is left unchanged; if the S trigger is zero, the pattern character is replaced by the fill character.

If the pattern character is either a digit-select or a significance-start character, the source digit is examined. The source digit replaces either of these pattern characters if the S trigger is one or if the source digit is non-zero. A non-zero digit inserted when the S trigger is zero will cause the S trigger to be set to one to indicate that the following digits are significant. If the S trigger and the source digit are both zero, the fill character is substituted for either the digit-select or significance-start character.

Source digit. When the source digit is stored in the result, it is expanded from the packed to the zoned format by attaching a zone. The zone code is 1111 in the binary coded decimal mode and 0101 in the ASCII mode.

The source digits are examined only once during an editing operation. They are selected eight bits at a time from the second operand field. The left-most four bits are examined first. The right-most four bits remain available for the next pattern character which calls for a digit examination. However, the right-most four bits are inspected for a sign code immediately after the left-mode four bits are examined.

Any of the plus-sign code 1010, 1100, 1110, or 1111 will set the S trigger to zero after the digit is inspected, whereas the minus-sign codes 1011 and 1101 will leave the S trigger unchanged. When one of these sign codes is encountered in the four right-most bits, these bits no longer are treated as a digit, and a new character is fetched from storage for the next digit to be examined.

A plus sign sets the S trigger to zero even if the trigger was set to one for a non-zero digit in the same source byte or by a significance-start character for that digit.

Fill Character. The fill character is obtained from the pattern as part of the editing operation. The first character of the pattern is used as fill character, and is left unchanged in the result field, except when it is the digit-select or significance-start character. In the later cases a digit is examined, and when non-zero, inserted.

Result Condition. To facilitate the blanking of all-zero fields, the condition code is used to indicate the sign and zero status of the last field edited. All digits examined are tested for the code 0000. The presence or absence of an all-zero source field is recorded in the condition code at the termination of the editing operation.

1. The condition code is made 0 for a zero source field, regardless of the state of the X trigger.
2. For a non-zero source field and an S trigger of one, the code is made 1 to indicate less than zero.
3. For a non-zero source field and an S trigger of zero, the code is made 2 to indicate greater than zero.

The condition-code setting pertains to fields as specified by the field-separator characters, regardless of the number of signs encountered.

For the multiple-field editing operations, the condition-code setting reflects only the field following the last field-separator character. When the last character of the pattern is a field-separator character, the condition code is made 0.

Condition Code

0 Result field is zero
1 Result field is less than zero
2 Result field is greater than zero

Move with Offset (MVO) Decimal SS-Format

(See A Next Page)

- Objective: The low-order four bits of the first operand are attached as low-order bits to the second operand, the second operand bits are offset by four bit positions, and the result is placed in the first operand location.
- F1 (1111 0001) is the Op code
- L1 is the field length of Operand 1
- L2 is the field length of Operand 2
- B1 D1 is the main storage address of Operand 1
- B2 D2 is the main storage address of Operand 2

Description (SLD 20-34)

The move operation consists of placing the second operand to the left of the adjacent to the low-order four bits of the first operand. The fields are processed right to left. If necessary the second operand is extended with high-order zeros. If the first operand field is too short to contain all bytes of the second operand, the remaining information is ignored. Overlapping fields may occur and are processed by storing a result byte as soon as the necessary operand bytes are fetched.

(See B Next Page)

Programming Example

In the following example, Move with Offset operation, the digits and signs are grouped in eight-bit bytes. The second operand has no sign. The result digits are shown under the first operand digits which they replace.

First Operand	77	88	99	0+	Second Operand	00	12	34	56
Result 1	01	23	45	6+					

Programming Note

The instructions for decimal arithmetic do not include shift instructions, since the equivalent of a shift can be obtained by programming. Programs for right or left shift and for an even or odd shift amount may be written with the Move with Offset instruction and logical move instructions.

An even right shift can be simulated by moving the sign code left.

An even left shift may be obtained by moving zeros to the right of the field, moving the sign code, and replacing the original sign with a zero.

An odd right shift can be simulated with the Move with Offset instruction.

An odd left shift may be obtained with the Move with Offset instruction, moving zeros to the right of the field, moving the sign code, and replacing the original sign code.

Condition Code

The condition code is not changed.

Validity Checking

The first and second operand bytes are not checked for valid codes.

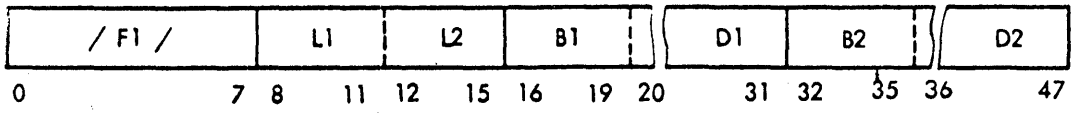
Pack (PACK) Decimal SS-Format

(See C Next Page)

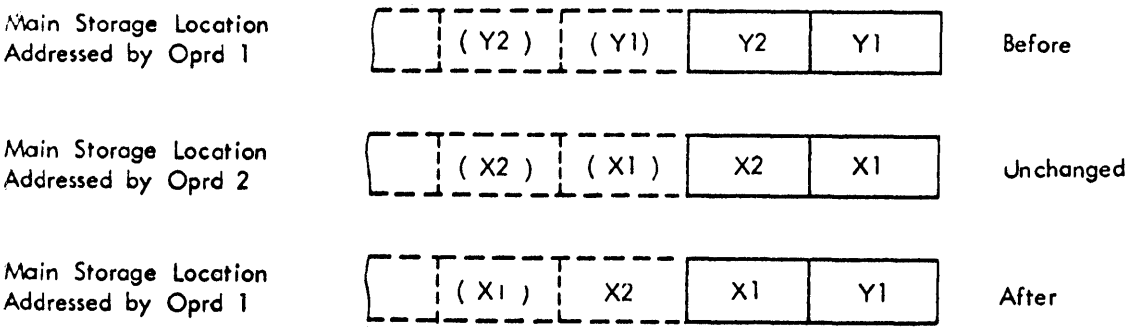
- Objective: The format of the second operand is changed from zoned to packed and the result is placed in the first operand location.
- The Op code is F2 (1111 0010).
- L1 is the field length of Operand 1.
- L2 is the field length of Operand 2.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-35)

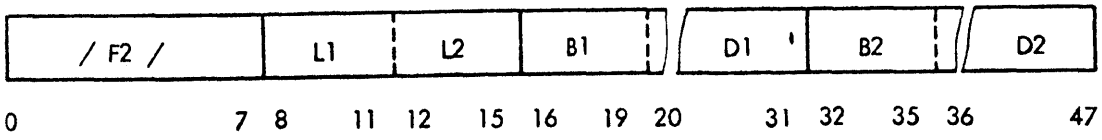
The second operand is assumed to have the zoned format. All zones are ignored except the zone over



A



B



C

the low-order digit which is assumed to represent a sign. The sign is placed in the right four bits of the low-order byte, and the digits are placed adjacent to the sign and to each other in the remainder of the result field.

The fields are processed right to left. If necessary the second operand is extended with high-order zeros. If the first operand field is too short to contain all significant digits of the second operand field, the remaining digits are ignored. Overlapping fields may occur and are processed by storing each result byte immediately after the necessary operand bytes are fetched.

(See A Next Page)

Programming Example:

In the following example of a Pack operation, the digits, signs, and zones are grouped in eight-bit bytes. The letter Z represents a zone code. The letter X represents an arbitrary four-bit code. The first operand occupies four bytes, the second operand occupies five bytes. The result bytes are shown under the first operand bytes which they replace.

First Operand XX XX XX XX Second Operand Z1 Z2 Z3 Z4 +5
Result 00 12 34 5+

(See B Next Page)

Condition Code

The condition code is not changed.

Unpack (UNPK) Decimal SS-Format

(See C Next Page)

- Objective: The format of the second operand is changed from packed to zoned and the result is placed in the first operand location.
- F3 (1111 0011) is the Op Code
- L1 is the field length of Oprd 1
- L2 is the field length of Oprd 2.
- B1 D1 is the main storage address of Oprd 1.
- B2 D2 is the main storage address of Oprd 2.

Description (SLD 20-37)

The digits and sign of the packed operand are placed unchanged in the first operand location using the zoned format. Zones with coding 1111 in the binary coded mode and coding 0101 in the ASCII mode are supplied for all bytes except the low-order byte, which receives the sign of the packed operand.

The fields are processed right to left. High-order zero characters are supplied if the second operand needs extension. If the first operand field is too short to contain all significant digits of the second operand, the remaining digits are ignored. The first and second operand fields may overlap, and are processed by storing a result byte immediately after the necessary operand byte is fetched.

(See D Next Page)

Programming Example

In following example of an Unpack operation, the digits, sign, and zones are grouped in eight-bit bytes. The letter Z represents the zone code 1111. The letter X represents an arbitrary four bit code.

First Operand XX XX XX XX XX Second Operand 01 23 45 6-
Result Z2 Z3 Z4 Z5 -6

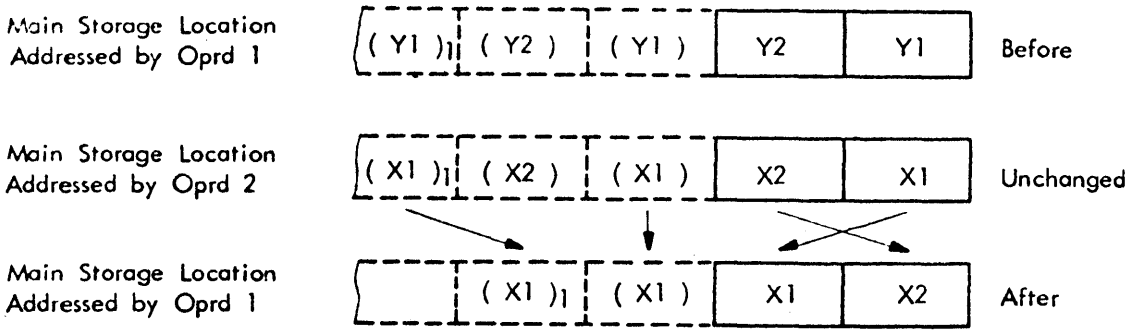
Condition Code

The condition code is not changed.

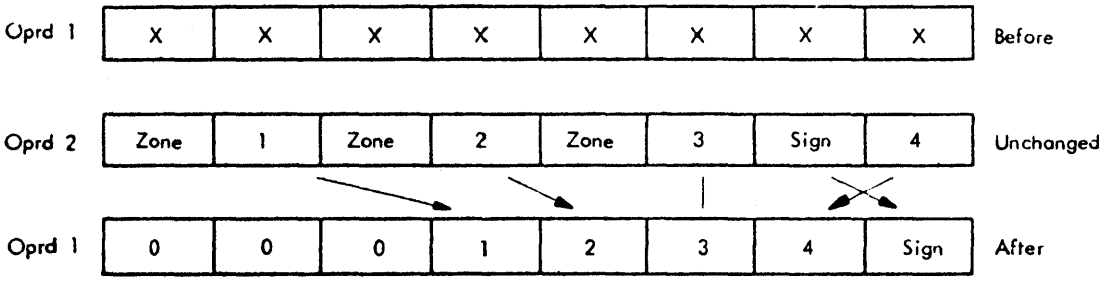
Validity Checking

The operand sign and digits are not checked for valid codes.

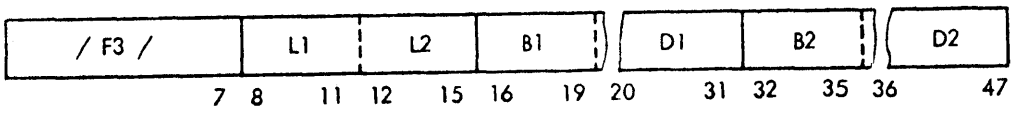
A



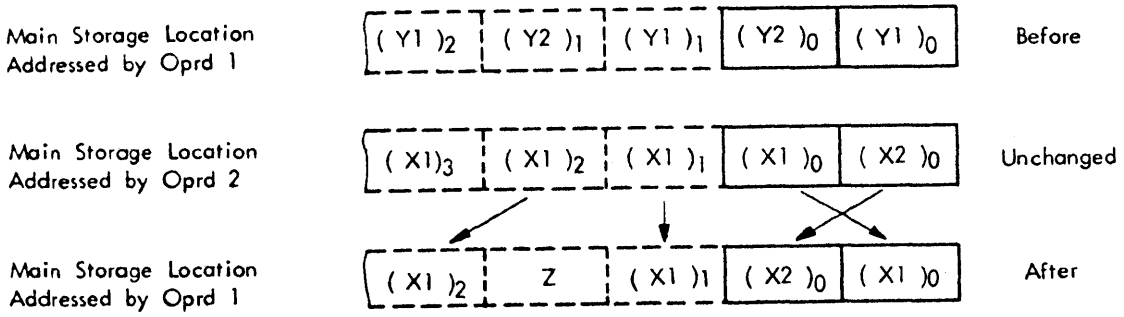
B



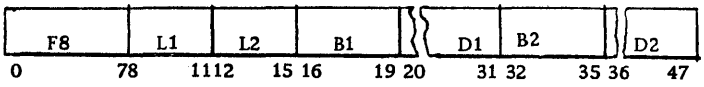
C



D



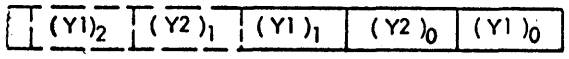
Zero and Add (ZAP) Decimal, SS Format



- The second operand is placed in the first operand location.
- The operand is equivalent to an addition to zero.
- The sign code is made 1100 for positive and 1101 for negative results in the binary coded decimal mode, and 1010 for positive and 1011 for negative results in the ASCII mode.
- A zero result is always positive.
- The Op code is F8 (1111 1000).
- L1 is the field length of Operand 1
- L2 is the field length of Operand 2
- B1 D1 is the main storage address of Operand 1
- B2 D2 is the main storage address of Operand 2

Description (SLD 20-36)

Main Storage Location
Addressed by Oprd 1 Addr

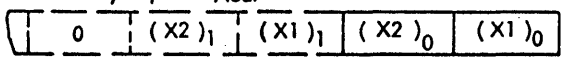


Before

Main Storage Location
Addressed by Oprd 2 Addr

Unchanged

Main Storage Location
Addressed by Oprd 1 Addr



After

Programming Note

Extra high-order zeros are supplied if needed. The first and second operand fields may overlap when the right-most byte of the first operand field is coincident with or to the right of the right-most byte of the second operand.

Program Error

When the length of the second operand (L2) is greater than the length of the first operand (L1) a specification error stop occurs. The instruction is not executed. The second operand is checked for valid sign and digit codes.

Programming Example

In the following sample Zero and Add operation, the digits and signs are grouped in eight-bit bytes. The letter X represents an arbitrary four-bit code. The first operand occupies five bytes, the second occupies three bytes. The condition code is set to 01 as a result of this zero and add operation.

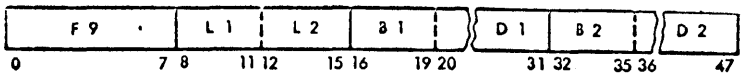
First Operand XX XX XX XX XX, Second Operand 38 46 0-
Result 00 00 38 46 0-

Condition Code

The condition code indicates the result of Zero and Add:

Result	Condition Code
Zero	0 0
Less than zero	0 1
Greater than zero	1 0

Compare Decimal (CP), Decimal, SS Format

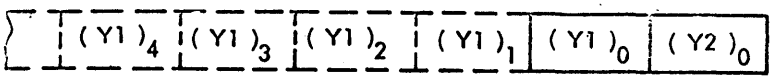


- Objective: The first operand is compared with the second operand and the condition code indicates the comparison result.
- Comparison is right to left, taking into account the sign and all digits of both operands.
- The Op code is F9 (1111 1001).
- L1 is the field length of Operand 1.
- L2 is the field length of Operand 2.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

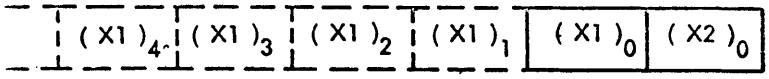
Description (SLD 20-41)

If the second operand field is shorter in length than the first operand field, the second operand field is extended with high-order zeros. A positive zero compares equal to a negative zero. Neither operand is changed as a result of the operation and overflow cannot occur. The first and second fields may overlap when their low-order bytes coincide. It is therefore possible to compare a number to itself.

Main Storage Position
Addressed by Oprd 1



Main Storage Position
Addressed by Oprd 2



Both Operands Remain Unchanged. Condition Code is Set by the Result of Complement Add of Both Operands.

Programming Note

Decimal Compare differs in several respects from Logical Compare. Decimal Compare is right to left. Signs, zeros, and invalid characters are taken into account and fields are extended when unequal in length. Also, the field length is restricted to sixteen 8-bit bytes, whereas Logical Compare permits fields up to 256 bytes.

Program Error Checking

When the length of the second operand (L2) is greater than the length of the first operand (L1) a specification error stop occurs. The instruction is not executed. All signs and digits are checked for validity.

Programming Example

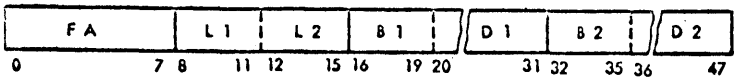
In the following sample Compare operation, the digits and signs are grouped in eight-bit bytes. The condition code is set to 01 as a result of this comparison.

First Operand 38 36 0+, Second Operand 54 32 1+

Condition Code

The condition code is made 00 if the operands are equal, 01 if the first operand is low, and 10 if the first operand is high.

Add Decimal (AP), Decimal, SS Format.

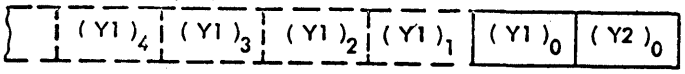


- Objective: The second operand is added to the first operand and the sum is placed in the first operand location. Addition is algebraic, taking into account sign and all digits of both operands.
- The sign of the result is determined by the rules of algebra. A zero sum is always positive. When high-order digits are lost because of overflow, a zero result will have the sign of the correct sum.

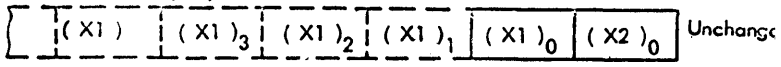
  The first and second operand fields may overlap when their low-order bytes coincide. It therefore is possible to add a number to itself.
- The Op code is FA (1111 1010).
- L1 is the field length of Operand 1
- L2 is the field length of Operand 2
- B1 D1 is the main storage address of Operand 1
- B2 D2 is the main storage address of Operand 2

Description (SLD 20-55)

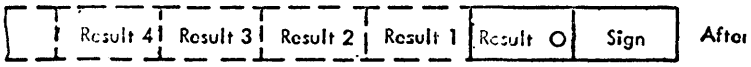
Main Storage Position
Addressed by Oprd 1



Main Storage Position
Addressed by Oprd 2



Main Storage Position
Addressed by Oprd 1



Program Error Checking. When the length of the second operand (L2) is greater than the length of the first operand (L1), a specification error stop occurs. The instruction is not executed.

All signs and digits are checked for validity. If necessary, high-order zeros are supplied for the second operand.

Programming Example. In the following sample Add operation the digits and signs are grouped in eight-bit bytes. The sum digits are shown under first operand digits which they replace. The condition code is set to 10 as a result of this addition. No overflow occurs.

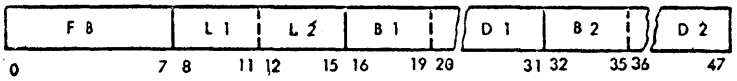
First Operand 38 46 0- Second Operand 50 80 5+
Sum 12 54 8+

Condition Code

The condition code is set according to the result of the Add Decimal operation:

Result	Condition Code
Zero	0 0
Less than zero	0 1
Greater than zero	1 0
Overflow	1 1

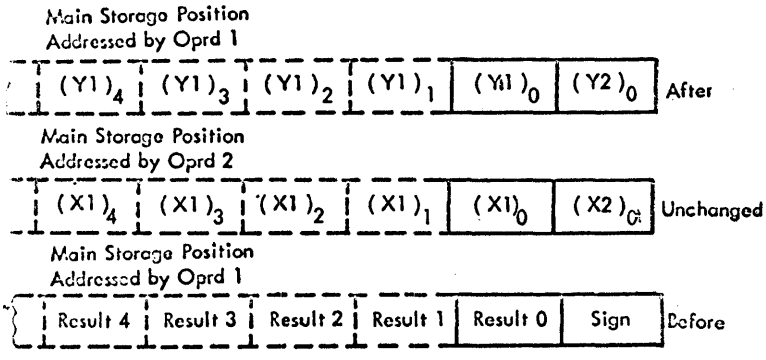
Subtract Decimal (SP), Decimal, SS Format



- Objective: The second operand is subtracted from the first operand and the difference is placed in the first operand location.
- Subtraction is algebraic, taking into account sign and all digits of both operands.
- The Op code is FB (1111 1011).
- L1 is the field length of Operand 1.
- L2 is the field length of Operand 2.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-39)

The Subtract instruction is identical to the Add instruction, except that the sign of the second operand is inverted prior to addition. The sign of the result is determined by the rules of algebra. A zero difference is always positive. When high-order digits are lost because of overflow, a zero result will have the sign of the correct difference.



Programming Note

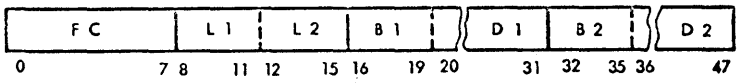
The operands of a Subtract operation may overlap when their low-order bytes coincide even when their lengths are unequal. This property may be used to make an entire field or the low order part of a field zero.

Condition Code

The condition code is set according to the result of the Subtract Decimal operation:

Result	Condition
Zero	0 0
Less than zero	0 1
Greater than zero	1 0
Overflow	1 1

Multiply Decimal (MP), Decimal, SS Format



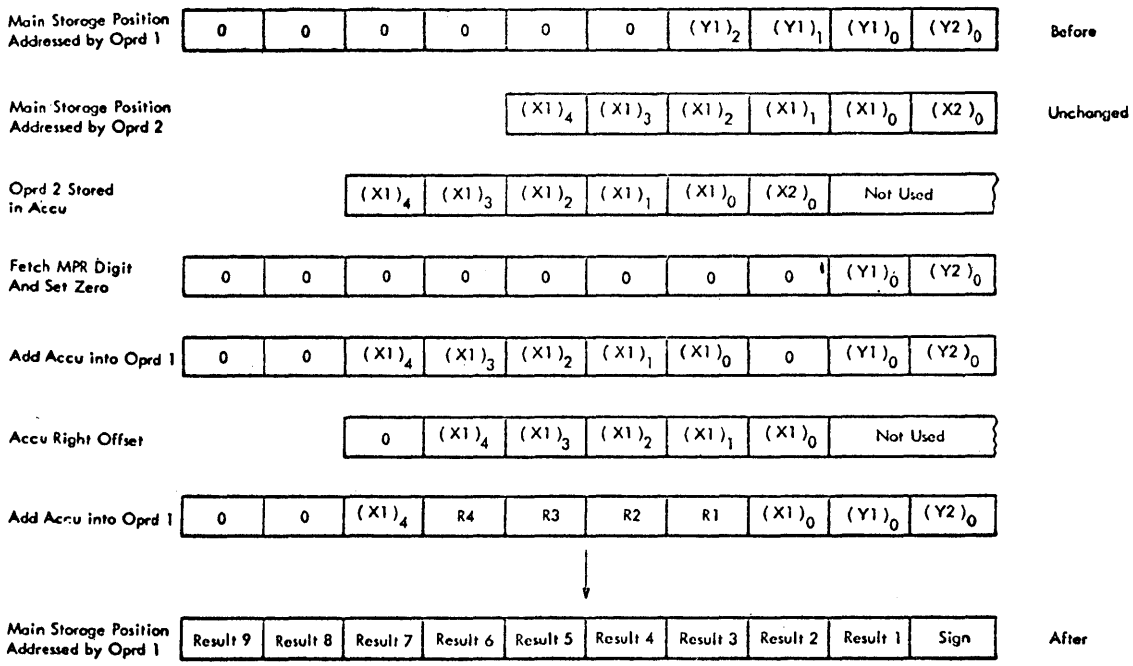
- Objective: The product of the multiplier (the second operand) and the multiplicand (the first operand) replaces the multiplicand.
- The Op Code is FC (1111 1100).
- L1 is the field length of Operand 1.
- L2 is the field length of Operand 2.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.
- All operands and results are treated as signed integers, right-aligned in their field.
- The sign of the product is determined by the rules of algebra from the multiplier and multiplicand signs, even if one or both operands are zero.

Description (SLD 20-51)

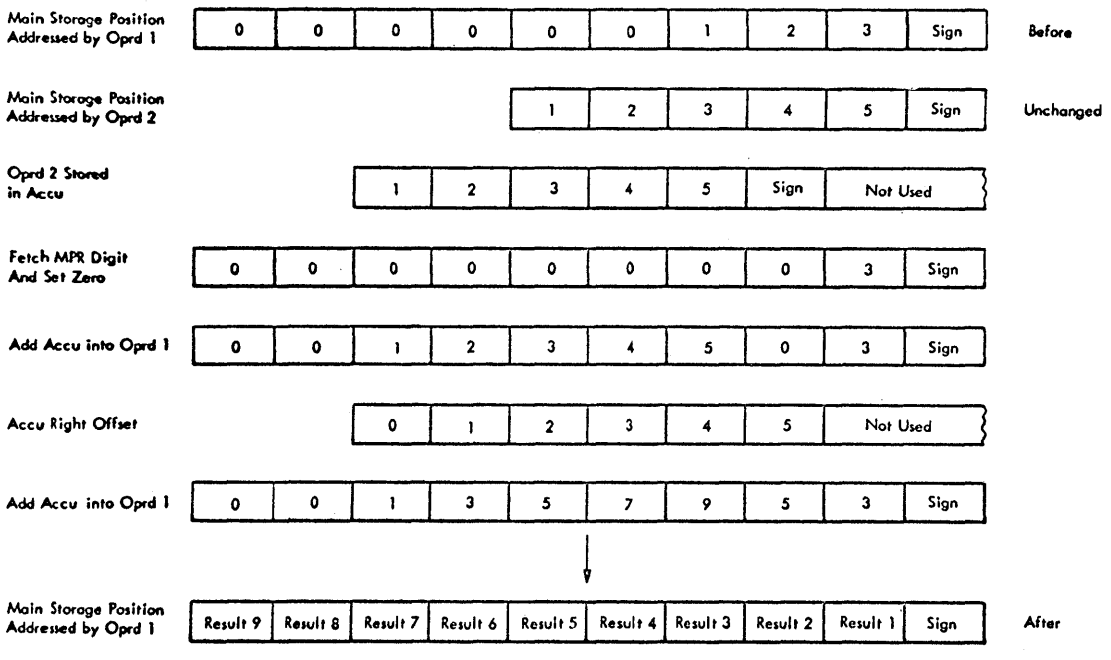
Since the number of digits in the product will be the sum of the number of digits in the operands, the multiplicand must have high-order zero digits for at least a field size which equals the multiplier size. Otherwise a data error occurs. This definition of the multiplicand field insures that no product overflow can occur. The maximum product size is thirty digits. At least one high-order digit of the product field will be zero. The multiplier and product fields may overlap when their low-order bytes coincide.

Programming Note

When the multiplicand does not have the desired number of leading zeros, multiplication may be preceded by a Zero and Add into a larger field.



Rn are the Results of the Single Additions



Program Error Checking. The multiplier size is limited to fifteen digits and sign, and must be less than the multiplicand size. If the length code L2 is larger than seven, or larger than or equal to the length code L1, the operation is not executed and a specification error stop occurs.

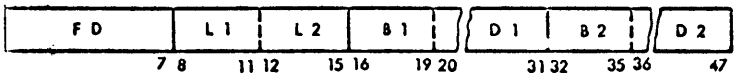
Programming Example. In the following example multiplication, the product digits and sign are shown below the multiplicand digits and sign which they replace. Digits and signs are grouped in eight-bit bytes.

Multiplicand 00 00 38 46 0- Multiplier 32 1-
Product 01 23 45 66 0+

Condition Code

The condition code remains unchanged.

Divide Decimal (DP), Decimal, SS Format



- Objective: The dividend (the first operand) is divided by the divisor (the second operand) and replaced by quotient and remainder.
- The dividend, divisor, quotient and remainder are all signed integers, right-aligned in their fields.
- The sign of the quotient is determined by the rules of algebra from dividend and divisor signs.
- The remainder has the same sign as the dividend.
- The above rules are true even when quotient or remainder is zero.
- The Op Code is FD (1111 1101).
- L1 is the field length of Operand 1.
- L2 is the field length of Operand 2.
- B1 D1 is the main storage address of Operand 1.
- B2 D2 is the main storage address of Operand 2.

Description (SLD 20-43, 20-45)

The quotient field is placed left-most in the first operand. The remainder field is placed right-most in the first operand field and has a size equal to the divisor size. Together the quotient and remainder occupy the entire dividend field. Therefore the address of the quotient field is the address of the first operand. The size of the quotient field in eight-bit bytes is L1 less L2 and the length code for this field is one less – (L1-L2-1). The divisor and dividend fields may overlap only if their low-order bytes coincide.

Programming Notes

The maximum dividend size is thirty digits and sign. Since the smallest remainder size is one digit and sign, the maximum quotient size is twenty nine digits and sign.

The condition for a divide check can be determined by a trial subtraction. The left-most digit of the divisor field is aligned with the left-most-but-one digit of the dividend field. When the divisor, so aligned, is less than or equal to the dividend, a quotient overflow is indicated.

Program Error Checking

The operation is not executed and a specification error stop occurs when the divisor length code is larger than seven, or larger than or equal to the dividend length code.

A divide check occurs when the quotient is larger than the number of digits allowed for it, or if the dividend does not have at least one leading zero. In that case the operation is not executed and a decimal divide error stop occurs. Divisor and dividend remain unchanged in their storage locations.

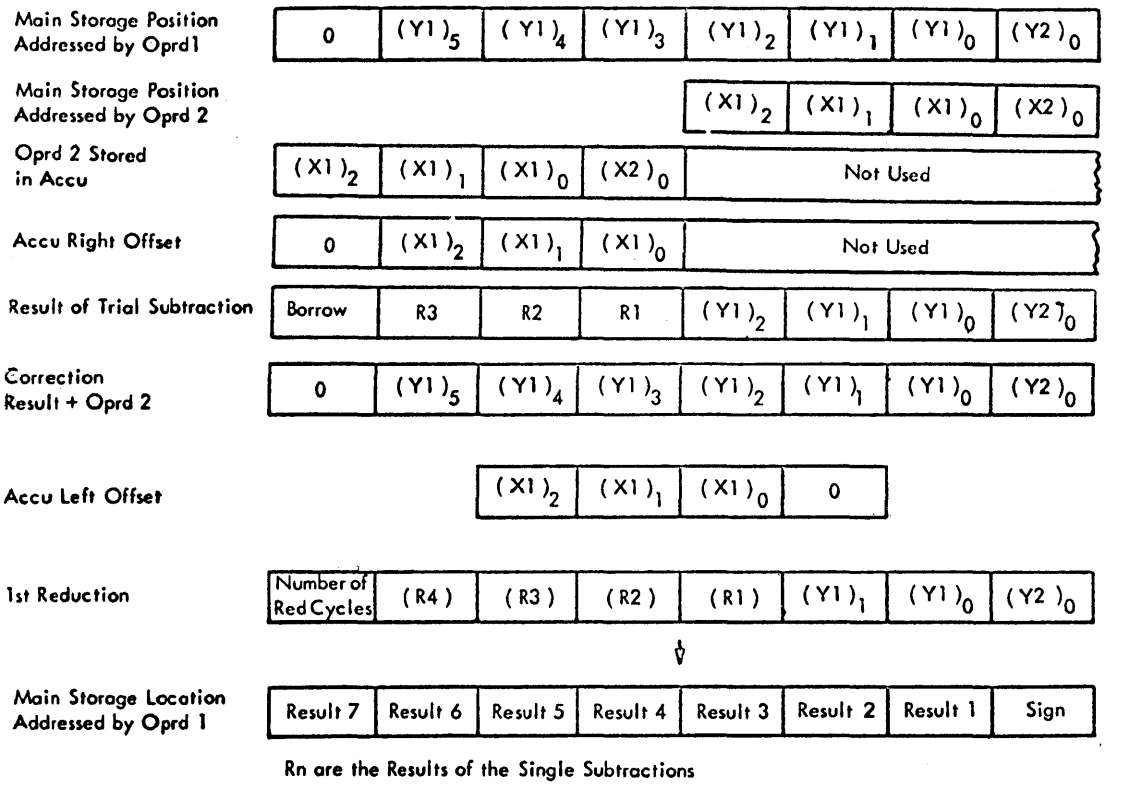
Programming Example

In the following example division, the quotient and remainder digits and signs are shown below the dividend digits and sign which they replace. Digits and signs are grouped in eight-bit bytes.

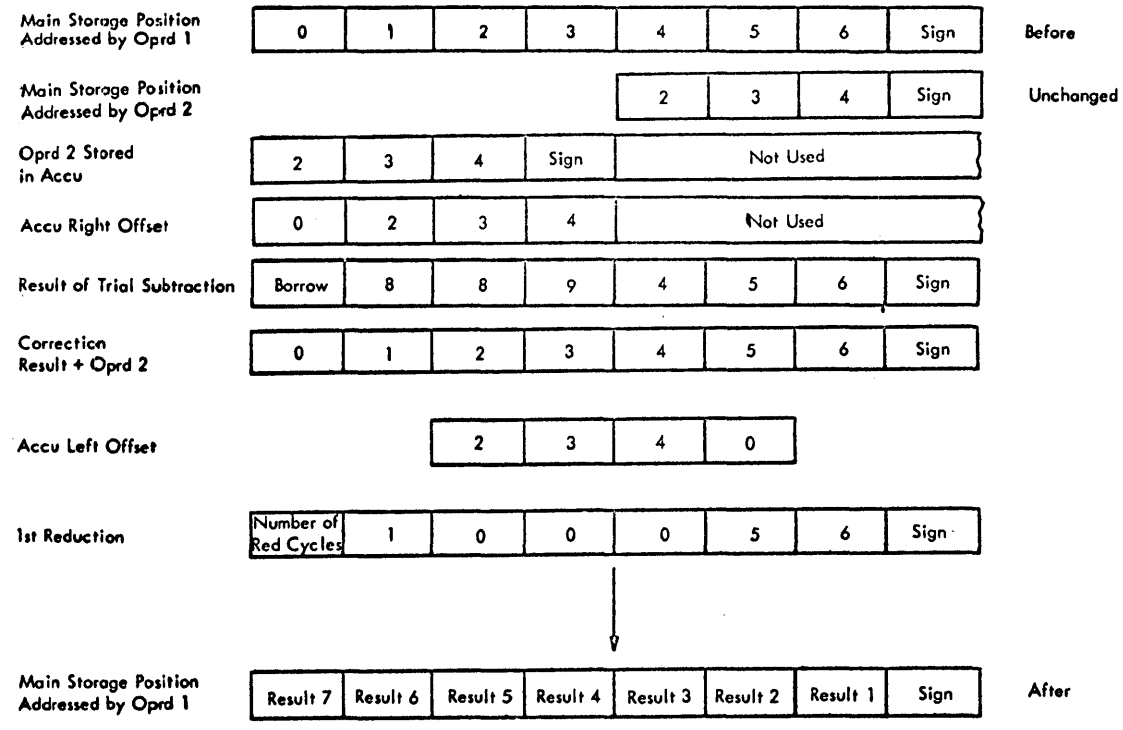
Dividend 01 12 45 67 8+ Divisor 32 1-
Quotient 38 46 0-
Remainder 01 8+

Condition Code

The condition code remains unchanged for division and overflow cannot occur.



Rn are the Results of the Single Subtractions



INPUT/OUTPUT OPERATIONS

- **Transfer of information from external sources to Main Storage and from Main Storage to external destinations are referred to as Input/Output operations.**
- **Input/Output operations are time shared to make maximum use of processing time.**
- **Input/Output operations consist of three phases: Instruction, Execution, and Service phases.**
- **Input/Output Service phases interrupt a program being executed.**
- **All Input/Output operations employ three types of instructions: Transfer I/O, Control I/O, Test I/O, and Branch.**

DATA

Data Formats

- **Input/output data is located in eight-bit bytes in Main storage in variable length fields.**
- **Data stored in Main storage can be in the zoned, fixed point or punched format, however a card-reader presents input-data in the zoned format and all output-data to a card punch or printer must be in the zoned format.**

Data Coding

- **Input data is translated from the code form of the input device to the Extended-Binary Coded Decimal Interchange Code employed by the CPU internally as the data is received.**
- **Output data is translated from the internal CPU to the zoned format as the data is transferred to the output device.**

ANY I/O REQUEST (Figure 3-40)

- Objective: Any I/O Request is a micro-program which handles the transfer of data from an I/O device to the CPU or vice-versa.
- The test for Any I/O Request is performed at intervals of approximately every 72 micro-instructions (This equals 42 usec) to see whether any I/O device is requesting service.
- When this test detects an I/O request, all data registers are cleared by storing their contents in Auxiliary Storage. The program is interrupted.
- The I/O device initiates its Service Phase during which the data of (for example) one column is read into DR-U and L.
- After a Service Phase, the program is continued until the next Any I/O Request test interrupts it again.
- For a given number of columns (Which are to be read in), the same number of I/O requests and respective Service Phases are required.
- More than one request for service may be present when the Any I/O request test is performed.
- Several requests are handled according to a priority sequence.
- The program is not continued until the last request has been acknowledged.

Description

Processing of a machine program is accomplished by any number of micro-instructions. There is a routine test interspersed in the chain of micro-instructions which is called Any I/O Request. This test is performed independent of the machine program after every 72 micro-instructions. It determines whether any I/O device is requesting service or not. It is necessary to repeat this test every 42 usec because the I/O units (Which run asynchronously) must be serviced often enough to avoid an accumulation of service requests.

1. Because the Any I/O Request routine represents an interruption of the micro-program in progress, an indicator bit is set in DR-U to enable the micro-program to continue where it was interrupted.
2. A test is made to see whether any I/O device is requesting service. If there is none, the indicator bits are tested to find out where the program is to be continued.
3. When an I/O request is present, a further indicator bit is set into DR-L.
4. The micro-program Housekeeping Forward sub-routine stores the contents of all data registers into Auxiliary Storage to save them. This is neccessary because the data registers are needed to transfer data from the I/O device to the CPU.

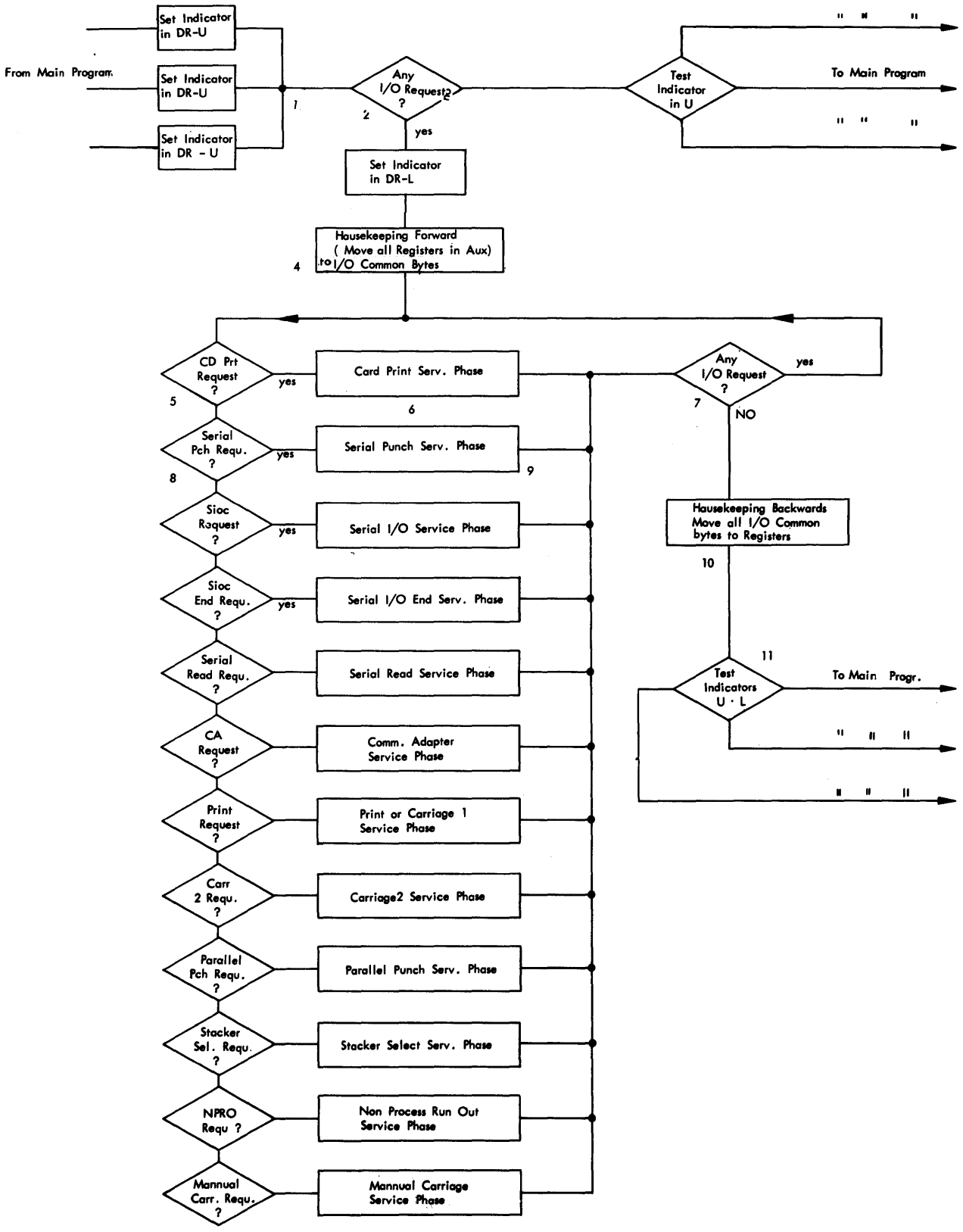


Figure 3-40 **Any I/O Request Loop**

5. Individual tests are made to determine exactly what type of request is present.
6. If there is a Card Print request (For example) present, the Card Print Service Phase is performed. During this Service Phase, the actual data transfer takes place.
7. When a Service Phase ends, the Any I/O Request test is performed once more. This is necessary because more than one request may be present.
8. If there is another request (For example Serial Punch), this request would be satisfied next and so on.
10. When the test in Step 7 shows that there is no request present, the Housekeeping Backward sub-routine brings back the data temporarily stored in the Common I/O bytes (Aux Positions /19/, /1A/, /1B/, /1C/) into the data registers.
11. The indicator bits in DR-U and L are tested to enable the micro-program to continue where it was interrupted. Any I/O Request comes up 42 usec later to take care of the requests that may have come up in the meantime.

TIME SHARING

- The Time Sharing feature allows processing operations in the CPU to be time shared with the transfer of data between Main Storage and the Input/Output devices.
- When an Input/Output device requests service, processing is interrupted only for the time required to transmit or accept Input/Output data.
- The Time Sharing feature can be controlled (on or off) by the operator via the Time Sharing switch on the CPU console.
- If the Time Sharing switch is off the machine carries out I/O operations in a sequential manner.

INPUT/OUTPUT PHASES

- There are three phases for each I/O operation (Figure 3-41).
- The Instruction (I) phase analyzes the meaning of an instruction.
- The I/O Execute Phase does not process data.
- The I/O Execute (E) phase sets an Execute Condition latch (one for each I/O unit) and initiates mechanical operation of an I/O unit.
- The I/O Execute phase is followed by the next I-phase.
- An I/O request detected during a Test I/O routine initiates a Service phase.

Description

The I/O operations of the Model 20 are to a large extend controlled by micro-programs. The Service phases of I/O operations are linked rather loosely together by the frequent interrogations made by the micro-program to determine if the need for an I/O data operation has arisen.

When a Service phase is terminated, the time between this Service phase and the next sequential Service phase of the same operation is used to accomplish other operations either in progress or about to begin (Figure 3-42). The other operations are I-phases, E-phases, or Service phases used to process data associated with other I/O units.

INSTRUCTION FORMATS

- To control a variety of I/O devices, three types of I/O instructions are provided.
  1. Transfer I/O
  2. Control I/O
  3. Test I/O and Branch
- These instructions generate signals required by the individual I/O devices.
- Table 3-3 shows the instruction formats employed by S/360-20 Input/Output Operations.

Transfer I/O

- A Transfer instruction governs the transfer of data between Main storage and the I/O device.
- The Device Address (D.A) specifies the I/O device to which output data is to be transmitted or from which input data is to be received.
- The Function Specification (F.S.) specifies the input or output function to be performed on the I/O device addressed and also the particular component of the addressed device when required.
- The main storage location of the first byte in the input or output data field is derived from the contents of the B1 and D1 fields according to the rules for direct or effective address generation.

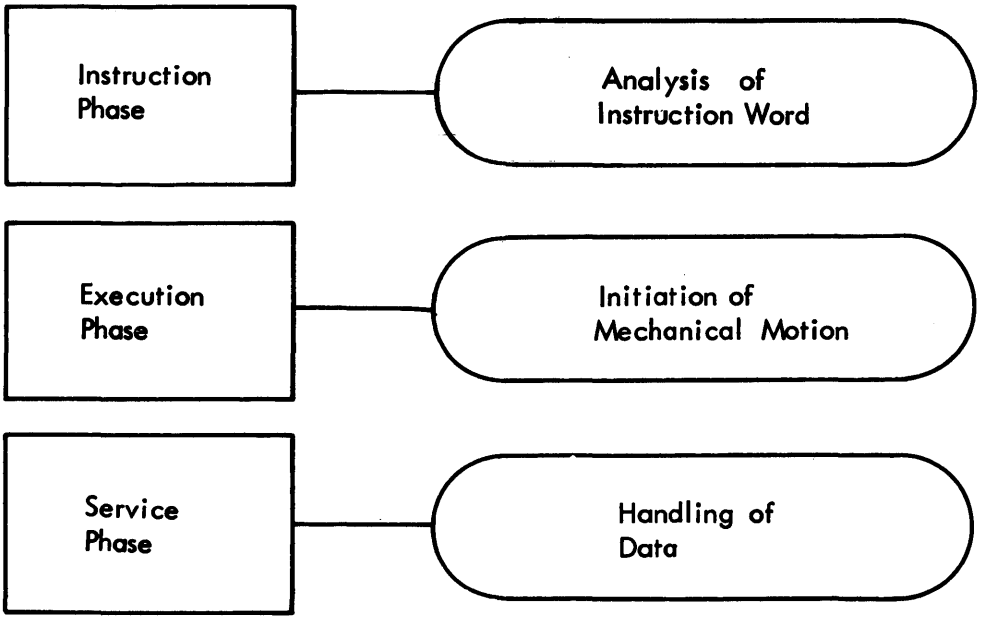


Figure 3-41 I/O Phase

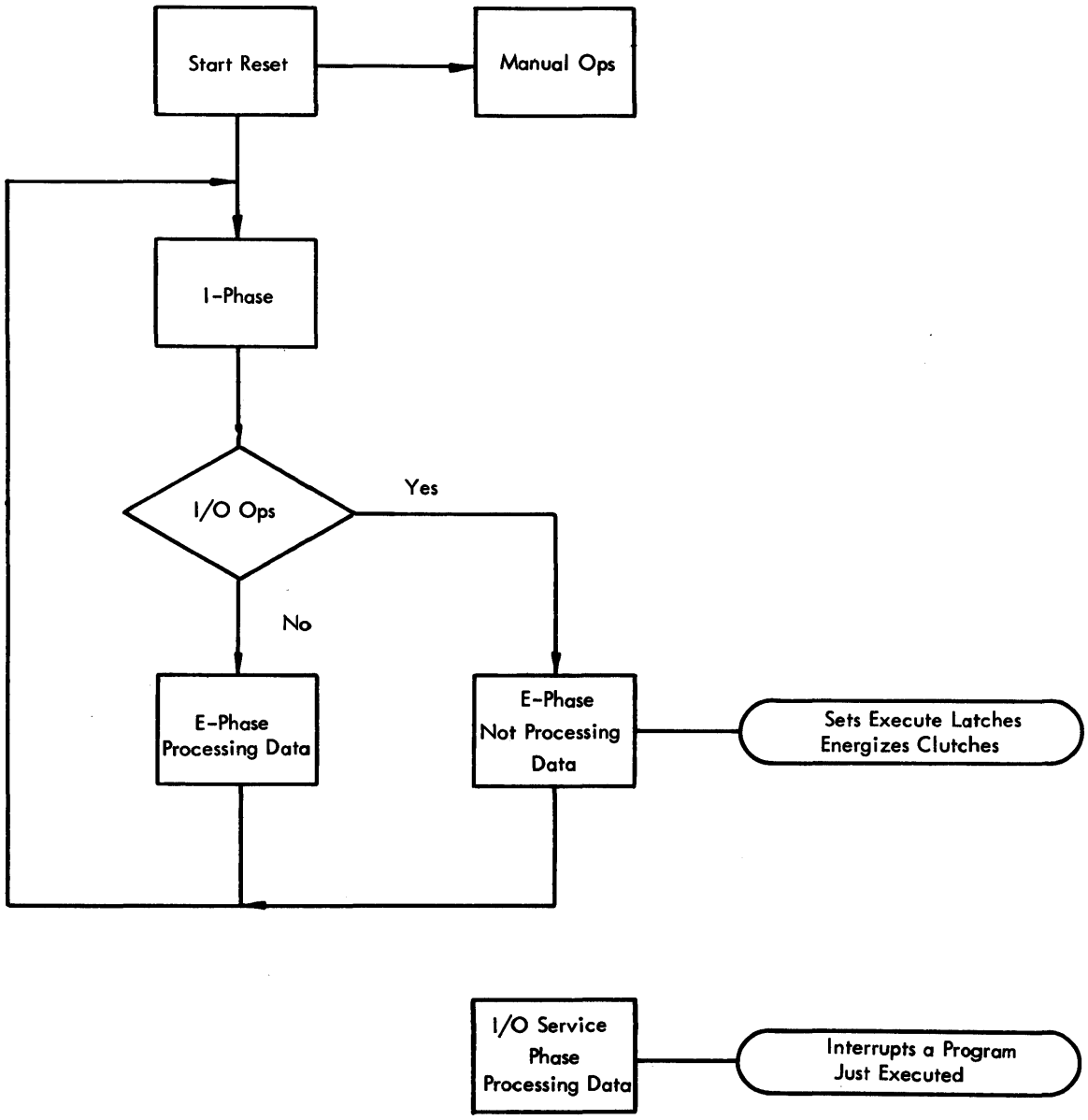


Figure 3-42 Micro-program Phases

Table 3-3. I/O Instruction Formats

Machine	Mnemonic Operation Code	Operand DA	Operand FS	Function
	XIO	1	2	Read Card
2501	XIO	1	10	Read Card, Column Binary
Card Reader	TIOB	1	0	Test Reader Busy
	TIOB	1	1	Test Reader Error
	TIOB	1	4	Test Last Card
	XIO	2	2	Read Primary Card
	XIO	2	3	Read Secondary Card
	XIO	2	4	Punch Primary Card
	XIO	2	5	Punch Secondary Card
	XIO	2	6	Punch and Feed Primary Card
	XIO	2	7	Punch and Feed Secondary Card
2560	XIO	2	10	Read Primary Card, Column Binary
Multi-Function	XIO	2	11	Read Secondary Card, Column Binary
Card Machine				
	XIO	2	0	*Card Print
	TIOB	2	0	Test Reader/Punch Busy
	TIOB	2	1	Test Reader/Punch Error
	TIOB	2	2	Test Card Print Busy
	TIOB	2	4	Test Last Card
	TIOB	2	5	Test Feed Error
	CIO	2	0	Primary Card Stacker Select
	CIO	2	1	Secondary Card Stacker Select
	CIO	2	2	Punch Card Stacker Select
	CIO	2	3	*Print Head Select
	XIO	2	2	Read Card
	XIO	2	4	Punch Card
	XIO	2	6	Punch and Feed
2520	TIOB	2	0	Test Reader Busy
Card Read	TIOB	2	1	Test Reader Error
Punch	TIOB	2	2	Test Punch Busy
	TIOB	2	3	Test Punch Error
	TIOB	2	4	Test Last Card
	TIOB	2	5	Test Feed Error
	CIO	2	0	Stacker Select
	XIO	4	0	Print
	XIO	4	1	Print and Space Suppress
	TIOB	4	0	Test Printer Busy
	TIOB	4	1	Test Printer Error
	TIOB	4	2	Test Channel 9
	TIOB	4	3	Test Channel 12
	TIOB	4	4	*Test Channel 9 (upper)
	TIOB	4	5	*Test Channel 12 (upper)
2203	CIO	4	4	Immediate Space
Printer	CIO	4	5	Immediate Skip
	CIO	4	6	Delayed Space
	CIO	4	7	Delayed Skip
	CIO	4	8	*Immediate Space (upper)
	CIO	4	9	*Immediate Skip (upper)
	CIO	4	A	*Delayed Space (upper)
	CIO	4	B	*Delayed Skip (upper)
	CIO	4	C	*Immediate Space (both)
	CIO	4	D	*Immediate Skip (both)
	CIO	4	E	*Delayed Space (both)
	CIO	4	F	*Delayed Skip (both)

(See Figure a on Next Page)

- The field or record length of the input or output data in main storage is derived from the contents of the B2 and D2 fields according to the rules for direct or effective address generation.

NOTE: The field length specification for input or output data fields in main storage is the actual number of bytes in the field, whereas for variable field length processing operations the field length specification is the number of bytes extending beyond the first byte.

Control I/O

- A Control instruction directs an I/O device to perform a specified function (for example, select a stacker pocket or initiate a carriage skip).

(See Figure b on Next Page)

- The Device Address (D.A.) specifies the I/O device in which a control function is to be performed

- The Function Specification (F.S.) specifies the particular component (it may also specify the primary function of that component) in the I/O device addressed in which a control function is to be performed.

- A detailed specification of the control function to be performed is derived from the contents of the B2 and D2 fields, according to the rules for direct or effective address generation.

Test I/O and Branch

- A Test I/O and Branch instruction causes an inquiryto an I/O device for a particular condition (for example: reader busy, end of forms etc.); if the indicator tested is on, the program branches to the specified address.

(See Figure c on Next Page)

- The Device Address (D A) specifies the I/O device in which a condition is to be tested.

- The Function Specification (F.S.) specifies the particular condition or indicator to be tested in the I/O device addressed.

- If the condition tested in the addressed I/O device is on, the updated instruction address is replaced by the branch address derived from the B2 and D2 fields; otherwise, normal instruction sequencing proceeds with the updated instruction address.

COLUMN BINARY FEATURE

The Column Binary Feature enables all serial card reading I/O devices included as system components, to operate in the column Binary mode as well as in the normal Extended Binary Coded Decimal mode.

In the Column Binary mode of operation, two adjacent bytes of main storage are employed to store one card column of data as shown below:

0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	Bits in main storage
-	-	12	11	0	1	2	3	-	-	4	5	6	7	8	9	Card Column

When reading in the Column Binary mode, bits 0 and 1 will be set to 0 in all bytes in the input field;

The Column Binary mode of operation is specified by an "8" bit (bit No. 12) in the Function Specification of the Transfer I/O instruction for serial card reading devices.

The normal Extended Binary Coded Decimal mode of operation is specified by the absence of the "8" bit in the Function Specification, (Table 3-4).

Table 3-4. 2560 and 2501 Functions

2560 MFCM

Device Address	Function Specification	Function	Mode
2	2	Read Primary Card	EBCD
2	3	Read Secondary Card	EBCD
2	4	Punch Primary Card	EBCD
2	5	Punch Secondary Card	EBCD
2	6	Punch and Feed Primary Card	EBCD
2	7	Punch and Feed Secondary Card	EBCD
2	10	Read Primary Card	Column Binary
2	11	Read Secondary Card	Column Binary

2501 Card Reader

1	2	Read Card	EBCD
1	10	Read Card	Column Binary

is a list of Transfer I/O instructions by device for all Card I/O devices to which the Column Binary Feature is applicable:

Mnemonic X I O Format I/O

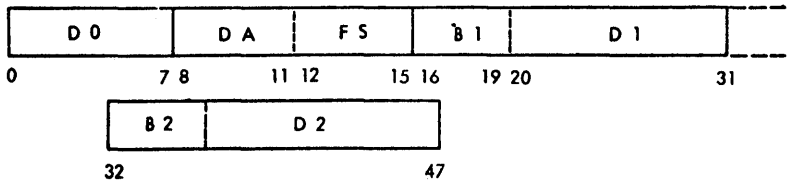


Figure a. Transfer I/O

Mnemonic C I O Format I/O

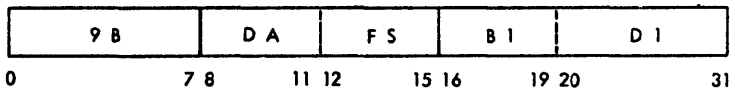


Figure b. Control I/O

Mnemonic TIOB Format I/O

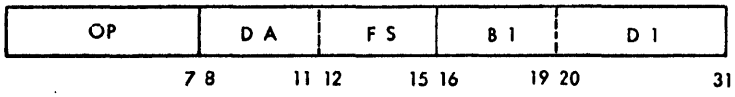


Figure c. Test I/O and Branch

CHECKING OPERATIONS

MODIFIER CHECK

- Check the output of the Modifier for correct (odd) parity.
- All data in the Bus latches are gated through the modifier.
- A parity error (even parity) turns on the Modifier Check latch.
- The Modifier Check Latch indicator is labeled "MOD" on the CE console.
- The Modifier Check latch turns on the Process Check latch.
- The Process Check latch being on causes the CPU to stop immediately (before the next ROS cycle starts).

Description (Figure 3-43)

An even Modifier output bit configuration (no bits, 2 bits, or 4 bits) causes a modifier check signal. This signal is ANDed with clock C to turn on the Modifier Check latch which lights its associated (MOD) lamp on the CE console. The Modifier Check latch ANDed with the same clock C pulse that turned it on, turns on the Process Check latch.

The Process Check latch can be turned on by seven other error conditions. When the Process Check latch is turned on it stops the CPU immediately (before the next ROS cycle starts) (Figure 3-44).

The Modifier Check latch is reset with Reset Condition (which is present when the Reset Key is pressed) or with Not Reset Condition and Clock C.

AHR CHECK

- Checks the contents of the AHR for correct (odd) parity.
- A parity error (even parity) turns on the AHR check latch.
- The AHR check latch indicator is labeled "AHR" on the CE console.
- The AHR check latch turns on the Process Check latch.
- The Process Check being on causes the CPU to stop immediately (before the next ROS cycle starts).

Description (Figure 3-45)

An even bit configuration in AHR (no bits, 2 bits, or 4 bits) causes an AHR check signal. This signal is ANDed with clock C to turn on the AHR Check latch which lights its associated (AHR) lamp on the CE console. The modifier check latch ANDed with the same Clock C that turned it on turns on the Process Check latch.

The Process Check latch can be turned on by seven other error conditions. When the Process Check latch is turned on it stops the CPU immediately (before the next ROS cycle starts) (Figure 3-44).

The AHR Check latch is reset with Reset Condition (which is present when the Reset Key is pressed) or with Not Reset Condition and Clock C.

STAR CHECK

- Checks the contents of the STAR for (odd) parity.
- A parity error (even parity) turns on the STAR Check latch.
- The STAR Check latch indicator is labeled "STAR" on the CE console.
- The STAR Check latch turns on the Process Check latch.
- The Process Check latch being on causes the CPU to stop immediately before the next ROS cycle starts.

Description (Figure 3-46)

All even STAR bit configurations (no, 2, 4, 6, 8, 10, 12, or 14 bits) cause a STAR Check signal. This signal is ANDed with Clock C to turn on the STAR Check latch which lights its associated (STAR) lamp on the CE console. The STAR Check latch ANDed with the same Clock C that turned it on turns on the Process Check latch.

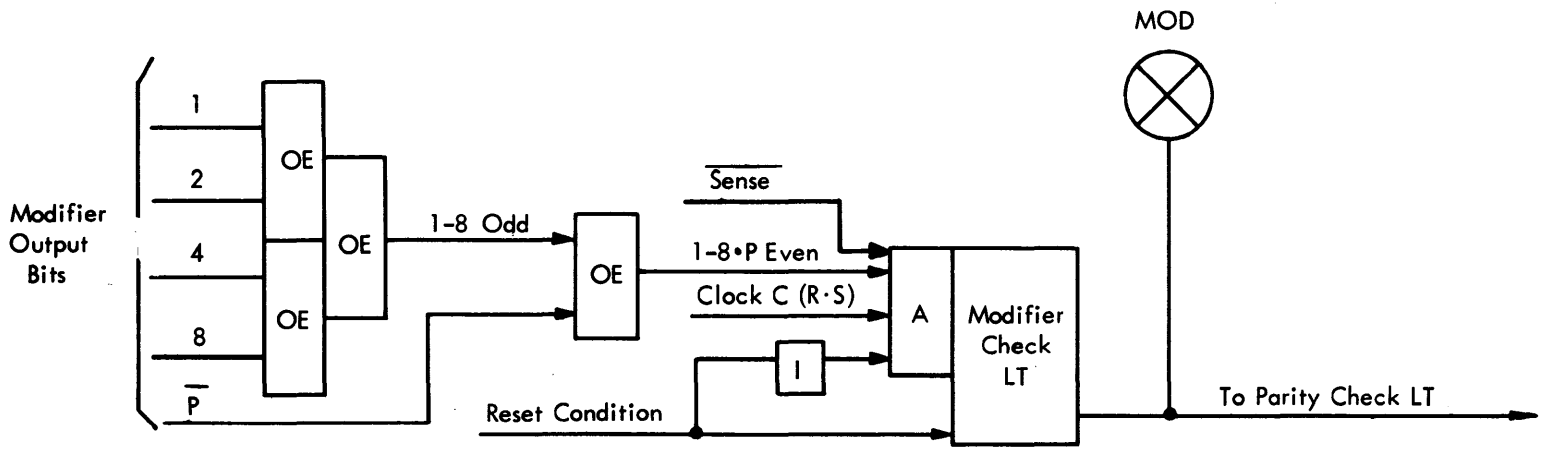


Figure 3-43 Modifier Check

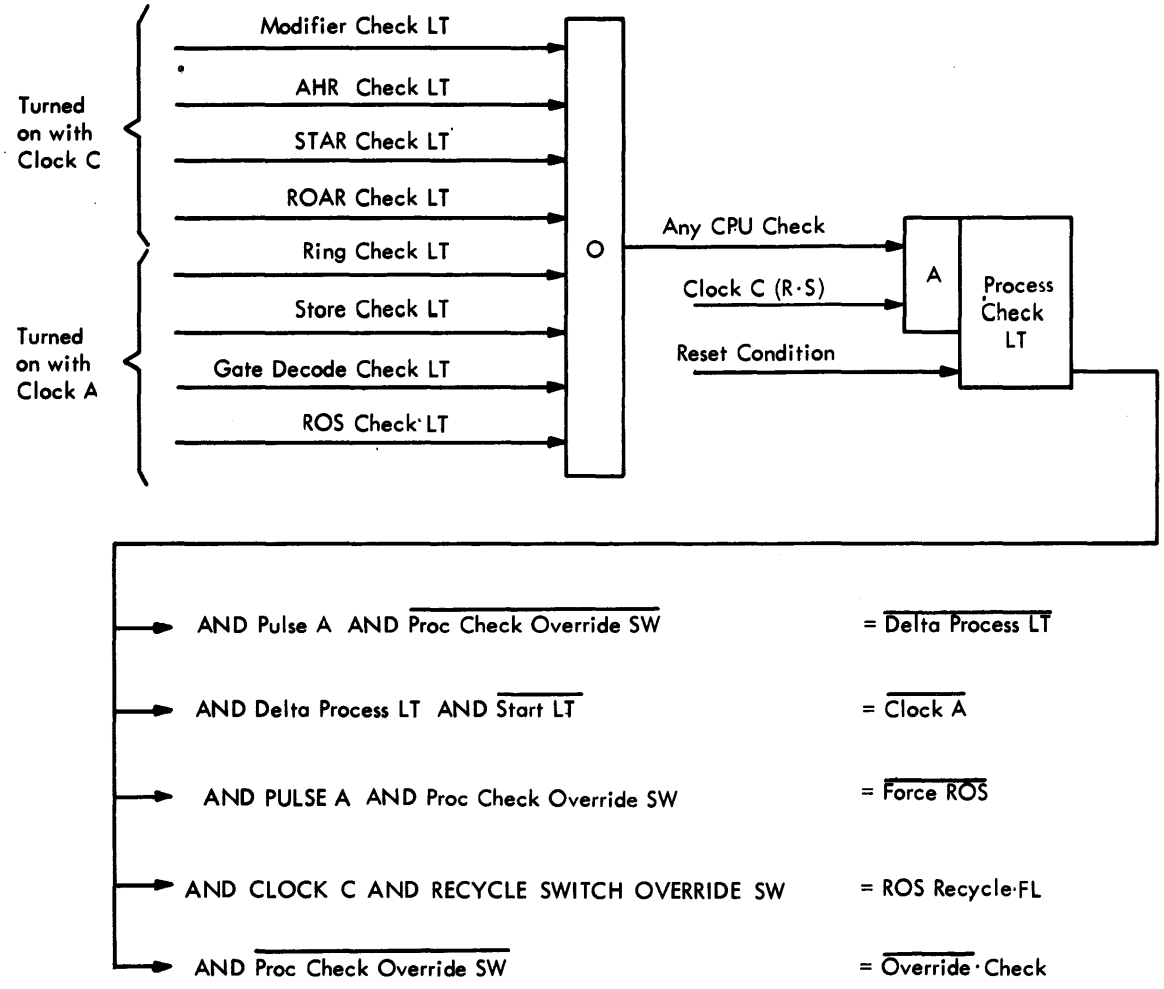


Figure 3-44 Process Check Latch

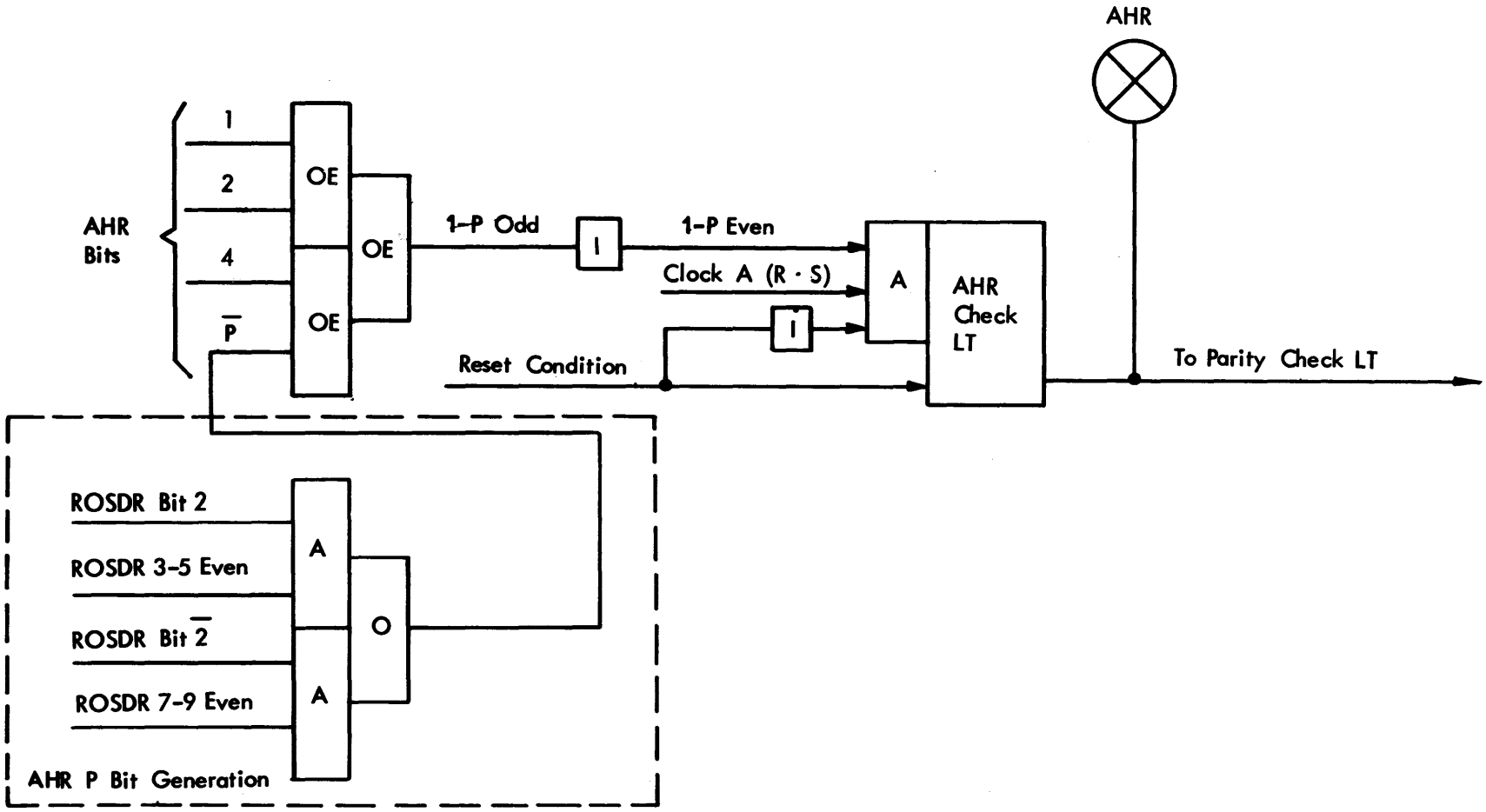


Figure 3-45 AHR Check

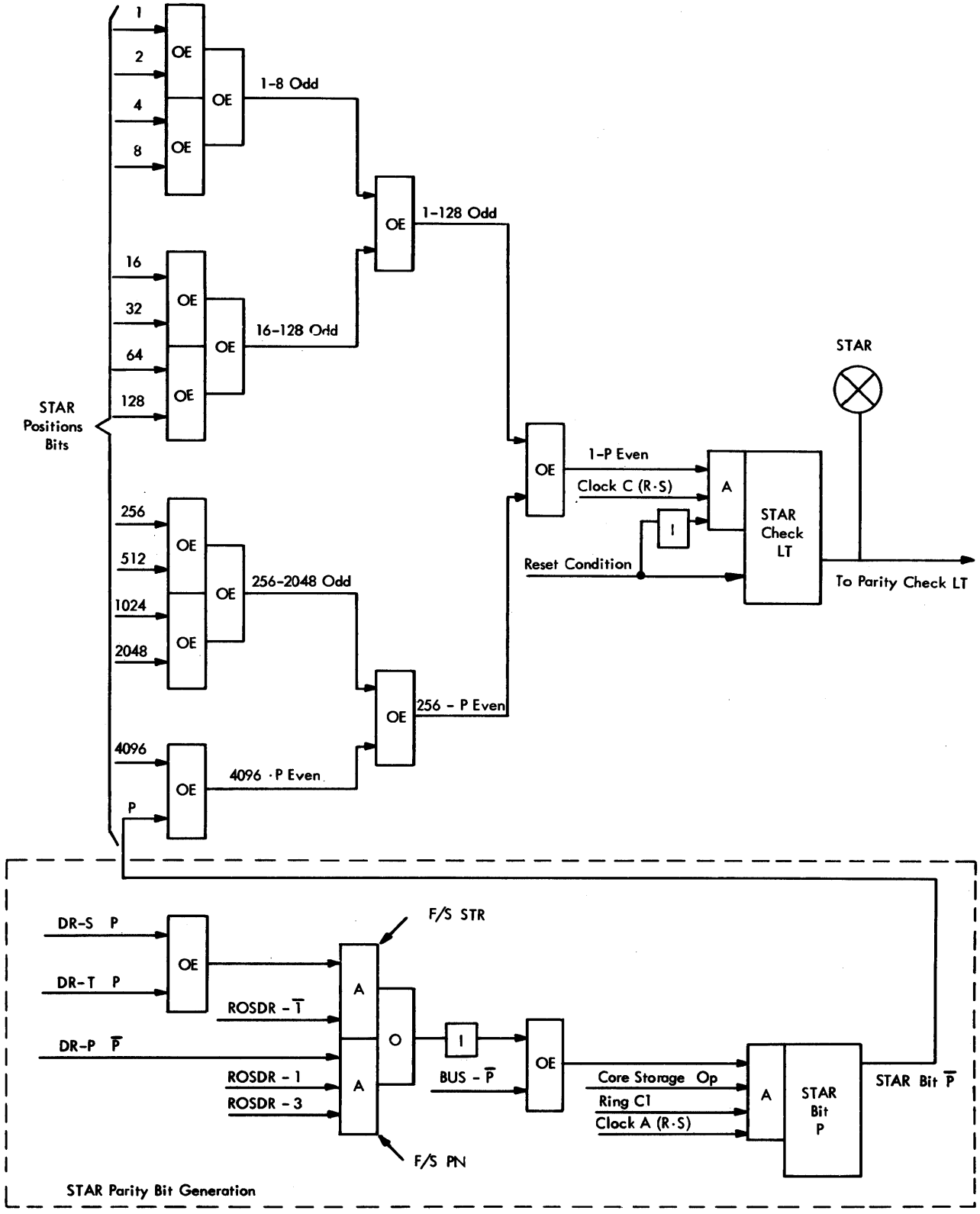


Figure 3-46 STAR Check

The Process Check latch can be turned on by seven other error conditions. When the Process Check latch is turned on it stops the CPU immediately (before the next ROS cycle starts) (Figure 3-44).

The STAR Check latch is reset with Reset Condition (which is present when the Reset Key is pressed) or with Not Reset Condition and Clock C.

ROAR CHECK

- The ROAR Check circuit compares the address in the ROAR (NSI address) with the Even ROS Address signal.
- The address in the ROAR is taken from the ROSDR pos. 10-21 and can be altered by a USE, INCR, or DECR micro-instruction, or by bit 16 and 17 in the ROSDR which generates bit E in the ROAR.
- To check the modified address in the ROAR the Even ROS Address circuit generates a signal "Even ROS Address" which is compared with the content of the ROAR.
- A check occurs when the ROAR address is even and no Even ROS Address signal is up or vice versa.
- The ROAR Check latch indicator is labeled "ROAR" on the CE console.
- The ROAR Check latch turns on the Process Check latch.
- The Process Check latch being on causes the CPU to stop immediately (before the next ROS cycle starts).

Description (Figure 3-47)

The contents of the ROAR used to address the ROS can have odd or even parity. (No parity bit is generated to create odd parity.)

All bits of the ROAR are gated to the ROAR Check circuitry, on exclusive OR network. The Even ROS Address signal is also gated into the same circuit (Figure 3-47).

An error occurs when the exclusive OR circuit output is active during Clock C. An output in the exclusive OR circuit is present when either the content of the ROAR is even and the Even ROS Address signal is not generated or the content of the ROAR is odd and the Even ROS Address signal is generated.

No ROAR check occurs when the contents of the ROAR has an even number of bits and the Even ROS Address signal is generated or vice versa.

The Even ROS Address signal is the output of the Even ROS Address latch (which is a latch in the ROS Check circuit). This latch is always on if an even (parity) ROS address is set into the ROAR.

The ROAR check lights its associated (ROAR) lamp on the CE console. The ROAR Check latch ANDed with the same Clock C that turned it on, turns on the Process Check latch.

The Process Check latch can be turned on by seven other error conditions. When the Process Check latch is turned on, it stops the CPU immediately (before the next ROS cycle starts) (Figure 3-44).

The ROAR Check latch is reset with Reset Condition (which is present when the Reset Key is pressed) or with Not Reset Condition and Clock C.

GATE DECODE CHECK

- Compares the output of Gate Decoder 1 with Gate Decoder 2.
- Gate Decoder 1 and Gate Decoder 2 should generate one output each and these should be identical.

  NOTE: No error is detected when gate Decoder 1 and gate Decoder 2 generates more than one gate as long as both generate the same multiple gates.
- An unequal output of gate Decoder 1 and gate Decoder 2 turns on the gate Decode Check latch.
- The Gate Decode Check latch indicator is labeled "Gate Dec" on the CE console.
- The Process Check latch being on causes the CPU to stop before the next ROS cycle.

Description (Figure 3-48)

Gate Decoder 1 and Gate Decoder 2 are two different circuits with the same input (ROSDR position 7-9). With each Clock A the outputs of Gate Decoder 1 and 2 are checked for equality. An unequal output AND Clock A turns on the Gate Decoder Check latch which lights its associated (Gate Dec) lamp on the CE console. The Gate Decoder check latch AND Clock C turns on the Process Check latch.

The Process Check latch can be turned on by seven other error conditions. When the Process Check is turned on it stops the CPU immediately

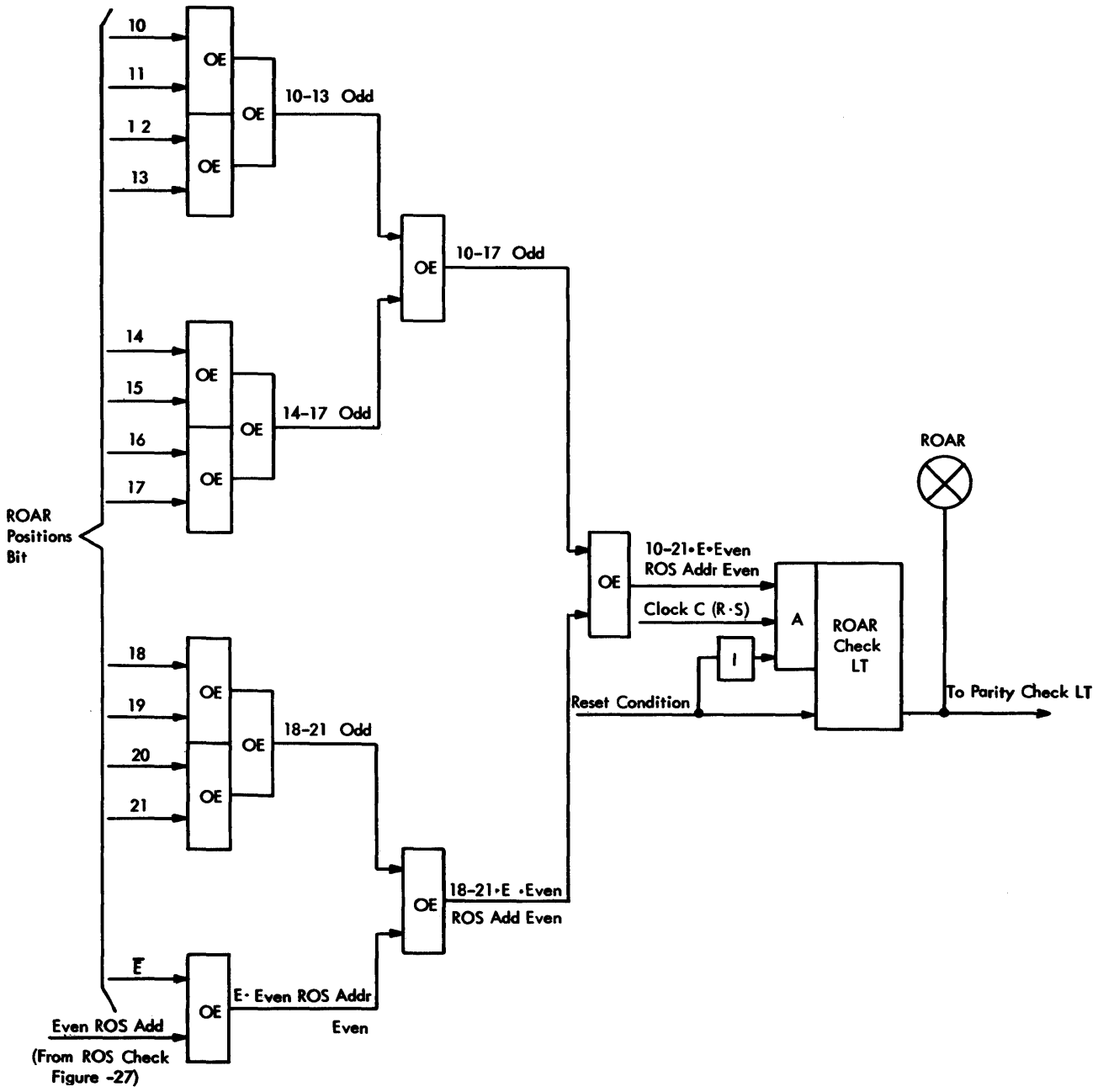


Figure 3-47 ROAR Check

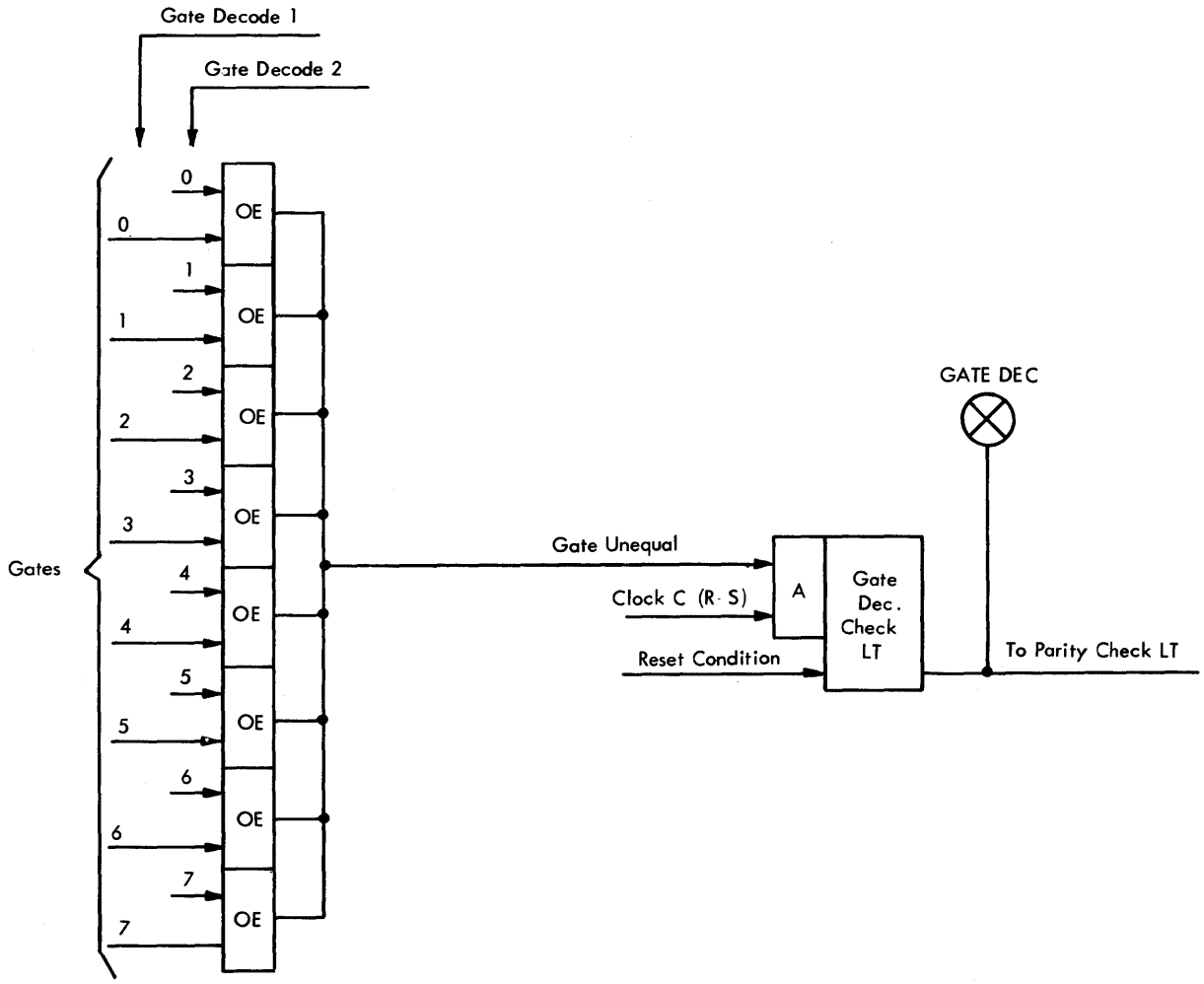


Figure 3-48 **Gate Decode Check**

(before the next ROS cycle starts) (Figure 3-44). The Gate Decode Check latch is reset with Reset Condition (which is present when the Reset Key is pressed).

NOTE: When Gate Decoder 1 and 2 generates more than one gate each, but the gates are equal, no error is detected.

C RING CHECK

- Checks the correct running of the C Ring.
- Only one of the three C Ring latches should be on at any one time.
- A check occurs when no C Ring latch is on or when two C Ring latches are on. (No error is detected if all 3 C Ring latches are on.)
- A Ring check turns on the Ring Check latch.
- The Ring Check latch indicator is labeled "Ring" on the CE console.
- The Parity check latch turns on the Process Check latch.
- The Process Check latch being on causes the CPU to stop before the next ROS cycle starts.

Description (Figure 3-49)

A Ring error is ANDed with Clock A to turn on the Ring Check latch which lights its associated lamp on the CE console. The Ring Check latch AND Clock C turns on the Process Check latch.

The Process Check latch can be turned on by seven other error conditions. When the Process Check latch is turned on it stops the CPU immediately (before the next ROS cycle starts) (Figure 3-44).

The Ring Check latch is reset with Reset condition (which is present when the Reset Key is pressed).

STORE CHECK

- Checks a byte for correct (even) parity while it is being stored into the main storage.
- A byte before being stored in Main Storage is located in the DR-U and L and each DR should contain an odd number of bits (odd parity).
- An error check occurs when the contents of DR-U and L taken together are odd (one even the other odd).
- The Store Check latch indicator is labeled "Store" on the CE console.
- The Store Check latch turns on the Process Check latch.
- The Process Check latch being on causes the CPU to stop before the next ROS cycle starts.

Description (Figure 3-50)

All bytes which are stored into the main storage (not AUX storage) are checked for even parity by taking the contents of DR-U and L together. Normally each DR contains odd parity. When an odd byte is stored the Store Check circuit generates a signal which is ANDed with Clock A and Store STR (Main Storage operations only) to turn on the Store Check latch. The Store Check latch lights its associated (Store) lamp on the CE console. The Store Check latch is ANDed with Clock C to turn on the Process Check latch.

The Process Check latch can be turned on by seven other error conditions. When the Process Check latch is turned on, it stops the CPU immediately (before the next ROS cycle starts) (Figure 3-44).

The Store Check latch is reset with Reset Condition (which is present when the Reset Key is pressed).

PROCESS CHECK LATCH

- The eight CPU checks are ORed to stop the CPU at the end of the cycle in which any hardware check occurs (Figure 3-44).
- The eight checks are: Modifier, AHR, STAR, ROAR, Ring, Store, Gate Decode, and ROS Check.
- When one (or more) of the checks is on, it is ANDed with Clock C to reset-set the Process Check latch.
- When the Process Check latch is on the CPU is stopped at the next Pulse A time (Clock A is suppressed).

Description

Figure3-44 shows the Process check circuits and the signals and latches which are dependent upon the Process Check latch being on.

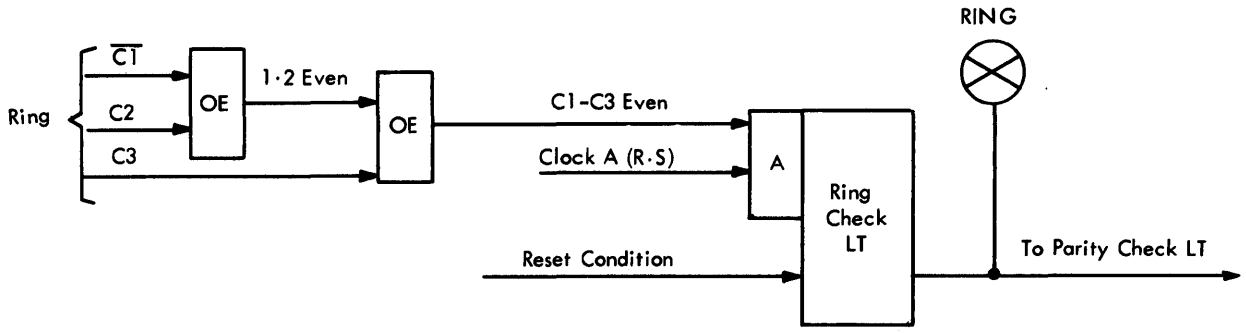


Figure 3-49 **Ring Check**

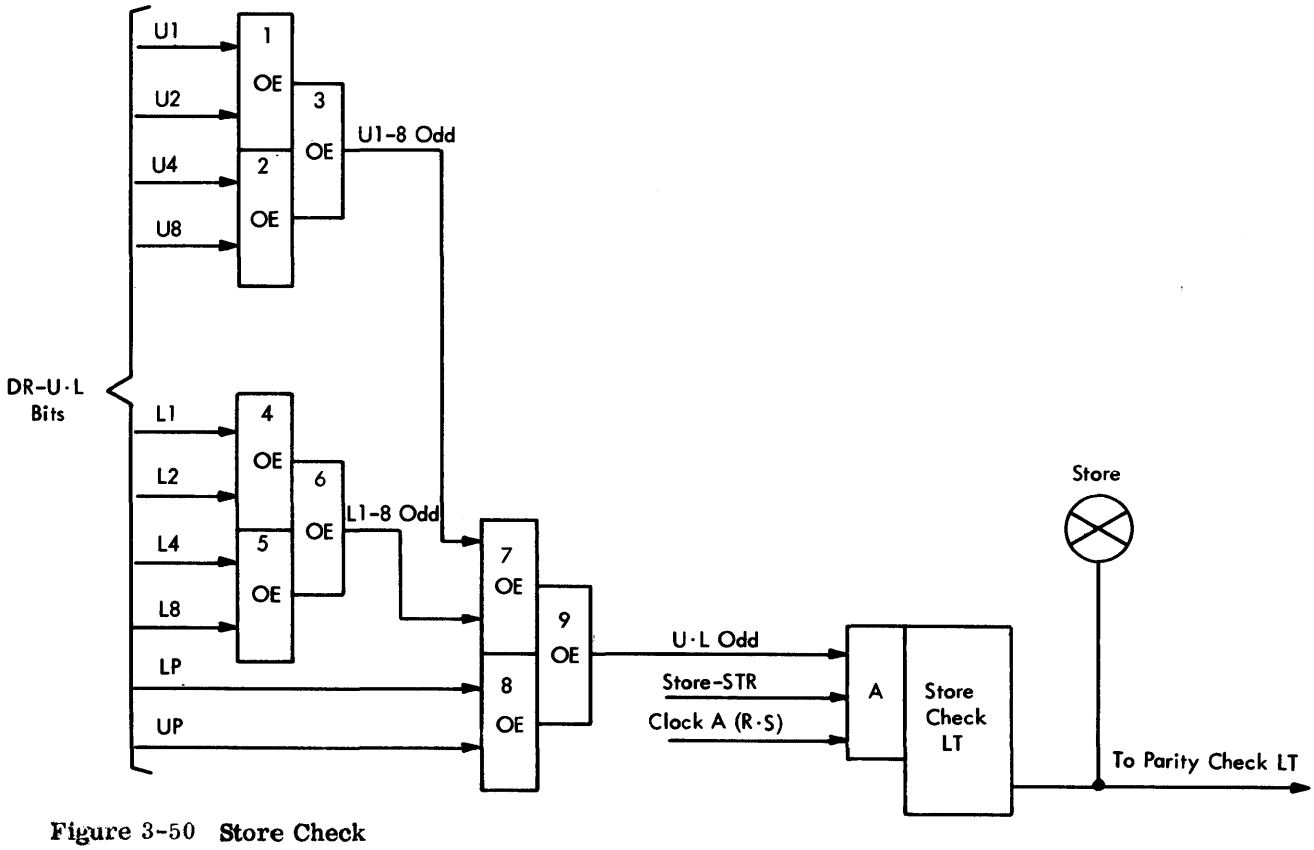


Figure 3-50 **Store Check**

ROS CHECK

- Checks two conditions 1 Broken Tape 2 ROS Parity.
  1. A Broken tape is detected when no bits are set into ROSDR pos. 1 through 5 when a micro-instruction is read out of the ROS.
  2. A ROS Parity error occurs when the 13 bit address of a micro instruction in ROAR combined with the 22 bits of the addressed micro-instruction in ROSDR, do not have odd parity.

- When the address of a micro-instruction is even, the addressed micro-instruction must be odd and vice versa.

- The Even ROS Addr (ERA) latch is on for an address with an even bit count and is off for an address with an odd bit count.

- When the addressed micro-instruction is in the ROSDR, ROSDR is checked for parity and must be odd if the Even ROS Addr latch is ON, and vice versa.

- Figure 3-51 shows the principle of ROS parity checking.

- A ROS error condition (Broken Tape or ROS Parity) turns on the ROS Check latch.

- The ROS Check latch indicator is labeled ROS on the CE console.

- The ROS Check latch turns on the Process Check latch.

- The Process Check latch being on causes the CPU to stop before the next ROS cycle starts.

Description (Figure 3-52)

Broken Tape. The ROS check latch is turned on by the Broken Tape signal and clock A. The Broken Tape signal is generated when no one bits are decoded in positions 1 through 5 of the ROSDR.

ROS Parity. At the same time that a 12-bit micro-instruction address (NSI address) is gated from the ROSDR to the ROAR it is also gated to the Even ROS Address circuitry where the odd/even bit count of the address is determined.

This address can be modified when it is set in ROAR, by a carry from an INCR or DECR micro-instruction or by a USE micro-instruction, or by the on/off condition of ROSDR bits 16 and 17. The same carry condition or USE micro-instruction (but not ROSDR bits 16 and 17) that modifies the address in ROAR is also used in the logic of the Even ROS Address circuits so that determination of the odd/even bit count is based upon the modified address (Figure 3-52).

NOTE: If, at Clock C time, the bit count is even the Even ROS address latch is turned on and if the bit count is odd, the latch is turned off.

The on/off condition of the Even ROS Address latch retains the bit count status of the address in ROAR so that, in the next cycle, when the 22-bit micro-instruction specified by the address is read out into ROSDR, the bit count status of the address can be combined with the bit count status of the micro-instruction. The combined bit count of a 13-bit address and the addressed 22-bit micro-instruction must be odd. Thus, if the bit count of the address is even (even ROS address latch ON) the bit count of the micro-instruction must be odd and vice versa.

If the combined bit count is even, the ROS Check latch is turned on and its associated CE console light (ROS) is turned on. The ROS Check latch AND Clock C turns on the Process Check latch.

It should be noted (Figure 3-52) that this system of parity checking makes dual use of parity checking circuitry.

Circuit Description for ROS Parity. The ROS Check latch is turned on by the ROS Parity Check signal and Clock A. The Parity Check signal is generated in the ROS Check circuit, when the bit count of the 13 bits of an address (micro-instruction address) combined with the 22 bits of its associated micro-instruction (addressed micro-instruction) do not have odd parity. Position 0 of the ROSDR, in which the 22-bit micro-instruction is located, contains the parity (P) bit.

As can be seen in Figure 3-53, at check time (Clock A), the 22 bits of the micro-instruction are in the ROSDR but the address of this micro-instruction has already been reset by the same Clock A. The bit count of the 13 bit address is therefore made during the previous cycle in which the address was available. The result of the bit count is stored in the Even ROS Address (ERA) latch. ERA latch is

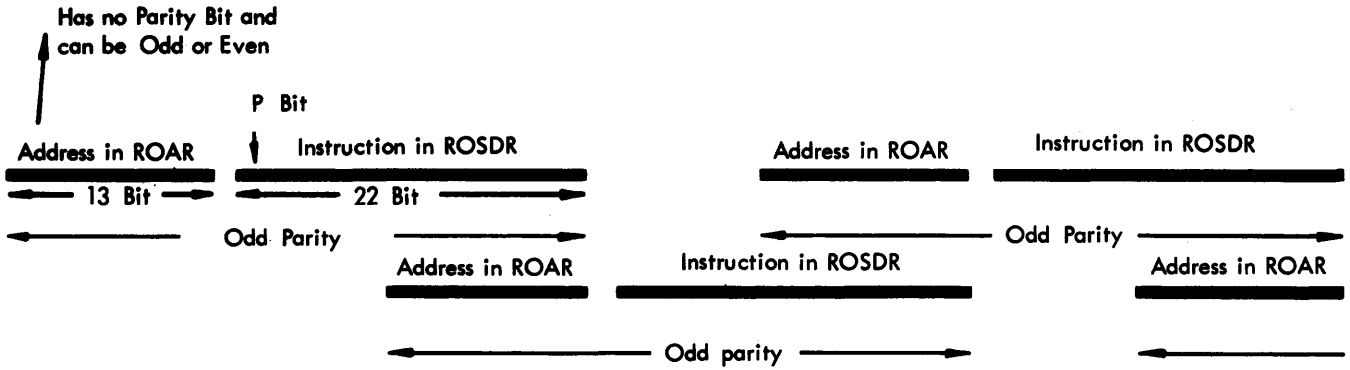


Figure 3-51 Principle of ROS Parity Checking

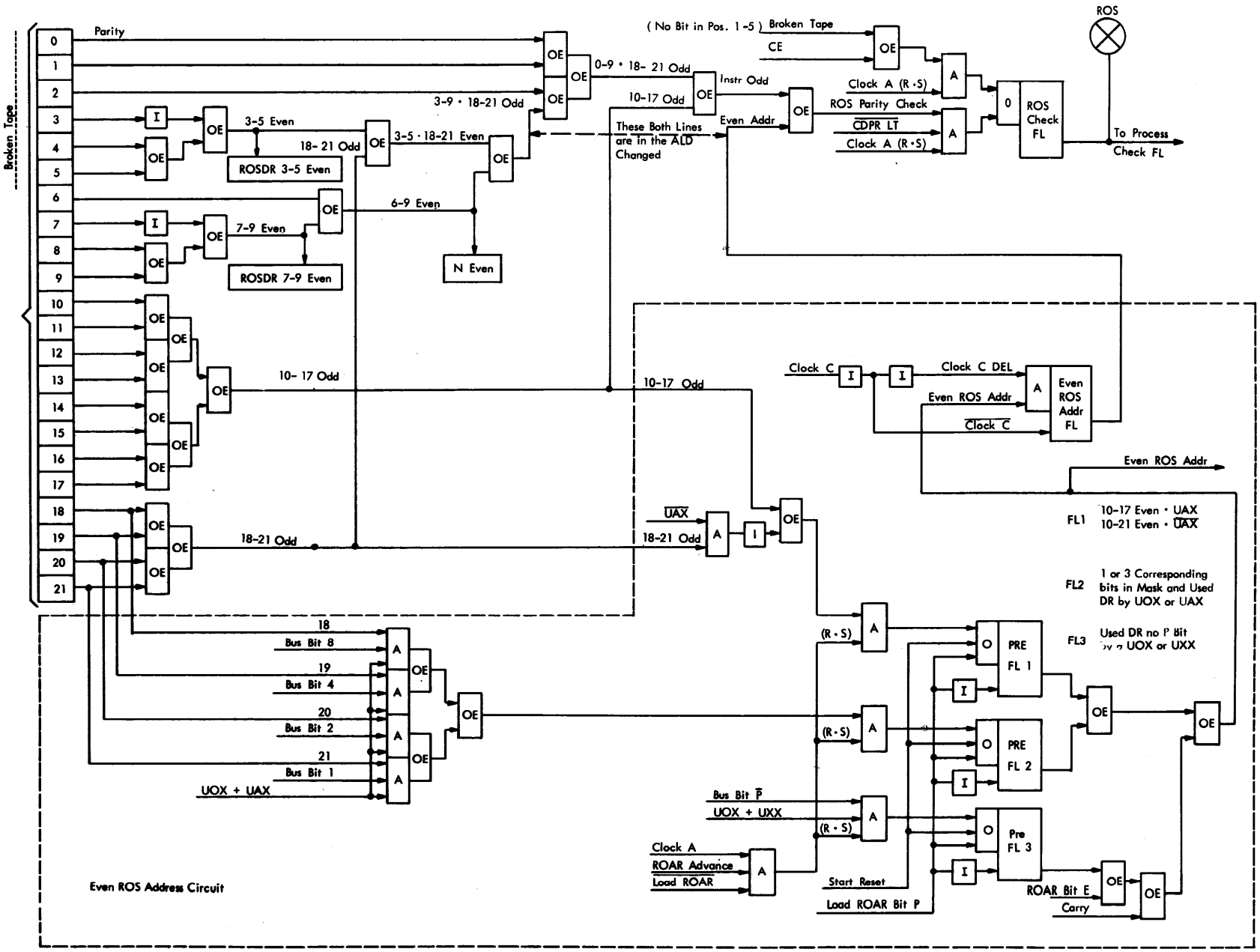
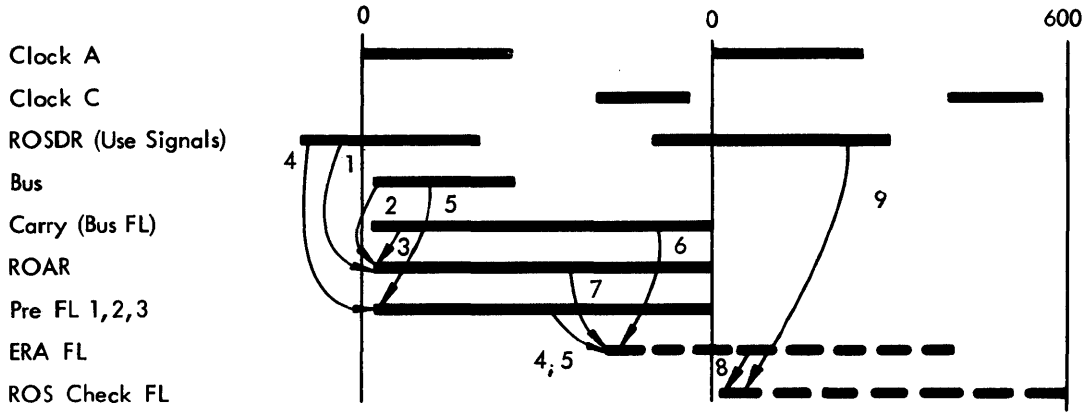


Figure 3-52 ROS Check



1 ROSDR Position 10-21 and Use Signal
2 Bus Bit 8,4,2,1
3 Carry
} Set ROAR

4 ROSDR Position 10-21 and Use Signal
5 Bus Bit P,8,4,2,1
6 Carry
7 ROAR Bit E
} Set ERA FL

8 ERA FL
9 ROSDR Position 0-21
} Set ROS Check FL

Figure 3-53 ROS Check Timing

turned on when the bit count of the address is even.

NOTE: Bit E in ROAR is set or reset depending upon ROSDR positions 16 and 17. The modification of the Even ROS Address circuitry is accomplished by using ROAR bit E and not ROSDR (Use signals an NSI address) and the ROSDR is reset (at time 300) before the ERA latch can be turned on with Clock C, three "Pre" latches store the various conditions. The "Pre" latches are labeled Pre FL 1, Pre FL 2, and Pre FL 3 and are reset set with Clock A.

Pre FL 1 is turned on when "Not UAX" and the bit count of ROSDR position 10 through 21 is even, or by a "UAX" and the bit count of ROSDR positions 10 through 17 is even.

Pre FL 2 is turned on when a UOX or UAX and either one or three corresponding bits are presented in the Mask and in the DR specified by the UOX or UAX.

Pre FL 3 is turned on by a UOX or UXX when the DR addressed by the UOX or UXX does not contain a P bit (Bus bit not P).

The three Pre latches together with a Carry and ROAR bit E are exclusive Ored and control the setting of the ERA FL. Any odd combination of the five conditions (Pre FL 1, 2, 3, Carry, ROAR bit E) indicates an even bit count in ROAR and turns on the ERA FL.

The 40 different situations which can occur to turn on the ERA latch are shown in Table 3-5.

The Broken Tape signal and the ROS Parity Check signal are ANDed with Clock A and turn on the Process Check FL.

The Process Check latch can be turned on by seven other error conditions. When the Process Check latch is turned on, it stops the CPU immediately (before the next ROS cycle starts) (Figure 3-44).

The ROS Check latch is reset with Reset Condition (which is present when the Reset key is pressed).

PARITY BIT GENERATION

STAR Parity Bit

When the contents of STAR are used to select a core storage position, the odd/even bit count of the STAR is checked for parity (odd parity is correct).

A special circuit (Figure 3-46)

generates the STAR parity bit (P bit) depending upon which register contents are set into STAR to address core storage.

The P bit is stored in the STAR P bit FL. All micro-instructions which select core storage (F/S N, F/S PN, and F/S STR) set a specific number of registers into STAR.

F/S N sets the N part (ROSDR pos. 6 through 9) into STAR. STAR P bit FL is turned on if the N part is even. The "N Bits Even" signal activates the Bus bit P, N Bit Even and Bus bit P up during a F/S N turns on the STAR P bit FL. "N Even" signal is determined by the ROS check circuit (Figure 3-51).

F/S PN sets the N part (ROSDR pos. 6 through 9) and the contents of DR-P into STAR. STAR P bit FL is turned on when the N part is even and DR-P has a P bit, or the N part is odd and DR-P has no P bit.

F/S STR sets the contents of DR-S, T, and R into STAR. STAR P bit FL is turned on when the sum of the P bits of DR-S, T, and R is odd.

Figure 3-46 includes the STAR parity bit generation circuit.

AHR Parity Bit

The contents of AHR is checked at clock C time for odd/even bit count (odd parity is correct).

A special circuit (shown in Figure 3-45), generates the AHR parity (P) bit depending upon which bit group of ROSDR is set into AHR.

When the bits of ROSDR positions 3 through 5 are set into AHR the P bit is set by the ROSDR 3 - 5 even signal which is detected by the ROS check circuit (Figure 3-51).

When the bits of ROSDR position 7 through 9 are set into AHR the P bit is set by the ROSDR 7 - 9 Even signal which is detected by the ROS check circuit (Figure 3-51).

STOP CONDITIONS

The Stop Indicator on the CPU console is turned off during normal operation of the central processing unit under control of the stored program. This indicator is on for all stop conditions of the central processing unit. Three types of stop conditions, other than Process Check, may occur during operation under stored program control:

1. Normal Stop (Stop condition latch ON)
2. Error Stop (Programming error stop)

In addition to the Stop Indicator, the following conditions are indicated in detail, by type in the P and I Register displays on the CPU console:

Table 3-5. Even ROS Address Table

Conditions → / Operations ↓	Pre FL 1		Pre FL 2	Pre FL 3	Carry	ROAR Bit E	ERA FL On
	$\overline{\text{UAX}}$ ROSDR 10-21 Even = X Odd = -	UAX ROSDR 10-17 Even = X Odd = -	UOX or UAX Mask and DR have 1 or 3 Corresponding Bits 1 or 3 = X 2 or 4 = -	UOX or UXX Used DR has No P Bit No P Bit = X P Bit = -	Yes = X	Yes = X No = -	
Normal (No Use or Carry)	X					-	YES
	-					-	NO
	X					X	NO
	-					X	YES
Carry	X				X	-	NO
	-				X	-	YES
	X				X	X	YES
	-				X	X	NO
UOX	X		-	-		-	YES
	-		-	-		-	NO
	X		X	-		-	NO
	-		X	-		-	YES
	X		-	X		-	NO
	-		-	X		-	YES
	X		X	X		-	YES
	-		X	X		-	NO
	X		-	-		X	NO
	-		-	-		X	YES
	X		X	-		X	YES
	-		X	-		X	NO
	X		-	X		X	YES
	-		-	X		X	NO
	X		X	X		X	NO
	-		X	X		X	YES
UAX		X	-			-	YES
		-	-			-	NO
		X	X			-	NO
		-	X			-	YES
		X	-			X	NO
		-	-			X	YES
		X	X			X	YES
		-	X.			X	NO
UXX	X			-		-	YES
	-			-		-	NO
	X			X		-	NO
	-			X		-	YES
	X			-		X	NO
	-			-		X	YES
	X			X		X	YES
	-			X		X	NO

	P Register	I Register
Normal Stop	0 0 0 0	0 0 0 0
Programming Error Stop	0 0 0 0	X X X X

NORMAL STOP CONDITIONS

A normal stop of the CPU occurs as a result of the following:

1. Operation of the Power On key.
2. Operation of the System Reset key.

   NOTE: After either of the operations above, all DR's are cleared to 0.

3. Operation of the Stop key.
4. An address stop in the Address Stop mode.
5. A Stop in the Single Instruction Mode.

   NOTE: After any of the three operations above, the DR's contain the values shown below.

Data Register	Contents
DR - U and L	NSI Op code
DR - E, S, T, and R	Address of the NSI Op Code

6. A programmed Halt operation (Op Code /99/).

   NOTE: After this operation, the DR's contain the values as shown.

Data Register	Contents
DR - U and L	Op Code /99/
DR - E, S, T, and R	Operand 1 address of the Halt instruction
DR - P and I	Cleared to /0/

Micro-Program Description (Stop Condition)

For all normal stops, the micro-program is described in the following sections.

1. Power On key and System Reset key in the Start Reset routine.
2. Stop key in the Stop key routine.
3. Address stop in the Address Stop routine.
4. Single Instruction stop in the Single Instruction routine.

PROGRAMMING ERROR STOP

A programming error stop occurs as a result of the following conditions and the DR's contain values as shown.

Data Register	Contents
DR - U and L	Op Code of the instruction in which the programming error occurs
DR - E, S, T, and R	Address of the Op Code in DR - U and L
DR - I	A value which identifies the different programming errors. (Seven different errors are possible)
DR - P	Cleared to /0/

A summary of the causes of programming errors is shown in Table 3-6.

Micro-Program Description (Programming Error)

When a programming error is detected in any micro-program, a binary value is moved into DR-I to specify the type of error. The micro-program then continues as described in the following paragraphs and as shown in step 110 in Figure 3-27a.

110. Move the Program Stop indicator from DR-I into DR-R, because DR-I is used later in the routine. Continue as shown in step 111.
111. Clear DR-T to /0/ for display purposes.
114. An address exchange moves the contents of DR-S, T, and R into the sub-register to save the contents of DR-T and R.
115. The Op code address of the instruction which caused the program error is moved from the I Recall Addr Reg Aux-storage position /10-11/ into DR-E, S, T, and R, by the Reg to ESTR subroutine.
116. The contents of DR-E, S, T, and R are moved into the I Addr Reg by the ESTR to Reg subroutine.

    The I Addr Reg now contains the Op code addresses of the erroneous machine instruction. This allows the customer to repeat the operation by pressing the Start key.
117. An address exchange moves the contents of the sub-registers into the DR's and the Op code address of the erroneous machine instruction into the sub-registers.
118. Move the Program Stop indicator from DR-R into DR-I for display purposes.
119. Move the contents of DR-T (/0/) into DR-P to clear it for display purposes.

120. An address exchange moves the Op code address of the erroneous instruction into 4K, S, T, and R.

63. The Op code addressed by the contents of 4K, S, T, and R is read out of Main Storage into DR-U and L. (The Op code is regenerated.)

10. A control 0 stops the CPU by resetting the Process latch. The DR's now contain the following values:

Data Register	Contents
U and L	The Op Code of the erroneous instruction
E, S, T, and R	The address of the Op code in DR - U and L
I	A number to indicate the type of program error
P	Cleared to /0/

I/O ATTENTION STOP

A CPU I/O Attention Stop occurs when an I/O instruction is decoded in the machine program and the addressed I/O unit is not able to perform an operation because some unusual condition is present. When an I/O Attention Stop occurs, the DR's contain the values shown below:

Data Register	Contents
U and L	Op code of the I/O instruction which can be executed
E, S, T, and R	Address of the Op code in DR - U and L
P	/F/ (I/O Attention Stop indicator)
I	Cleared to /0/

To quickly determine the I/O unit which caused the I/O attention stop, the following indicators lights are available on the CPU console:

Indicator	Unit
Card I/O 1	2501
Card I/O 2	2520/2560
Card I/O 3	1442-5
Print	1403 or 2203

Table 3-6 Cause of Programming Errors

Type of Error	Data Register Contents	Cause of Error
Invalid Op Code	/1/ in DR-I	An invalid Op code was decoded in the I-phase (45 Op codes are valid, out of 256 possibles)
Addressing Error	/4/ in DR-I	An I-address or an Operand address refers to the protected main storage area (Addresses lower than /0090/ = 144.)
	/5/ in DR-I	An I-address or an Operand address is outside the range of available storage (4K greater than /FFF/; 8K greater than /1FFF/; or 16K greater than /3FFF/. 1. In the RR format, the R1 field is /0/ through /7/, except the Op Code Branch on Condition /07/. 2. In the RR format, the R2 field is /0/ through /7/ except the Op Code Branch on Condition /07/ or Branch and Store /0D/. 3. In the RX format, the R1 field is between /0/ through /7/ except the Op Code Branch on Condition /47/.
Specification Errors	/6/ in DR-I	1. The Op Code address does not address a Boundary (even) main storage location. 2. In the RX format, the main storage address on all half word operations (Op Codes /40/, /48/, /49/, /4A/, /4B/, /4C/ or /4D/) does not address a boundary (even) main storage address. 3. In the RX format, the R2 field is not /0/. 4. In a Decimal Add /FA/, Decimal Subtract /FB/, or Decimal Compare /F9/ operation, the L2 field represents a value greater than the L1 field (Operand 2 length is greater than Operand 1 length). 5. In a Decimal Multiply /FC/ or Decimal Divide instruction, the L2 field is greater than 7, or is equal to or greater than the L1 field. 6. An I/O instruction is addressing an I/O unit which is not attached to the system. 7. In a transfer I/O /D0/ instruction, the field length is zero or greater than the correct number for the addressed I/O unit.
Data Error	/7/ in DR-I	1. A sign or digit value of an Operand in the Decimal operations Zero and Add /F8/, Add /FA/, Subtract /FB/, Compare /F9/, Multiply /FC/, or Divide /FD/ is incorrect or the Operand fields in these operations overlap incorrectly. 2. The multiplicand field (Operand 1) in a Decimal Multiply /FC/ operation has insufficient high-order zeros. 3. An invalid digit value is contained within the Operand 2 field of an Edit /DE/ operation.
Binary Overflow	/8/ in DR-I	A binary overflow error is detected at the end of the following binary arithmetic operations: 1. RR format Load Complement /13/, Add /1A/, and Subtract /1B/. 2. RX format Add /4A/ and Subtract /4B/
Decimal Divide Check	/B/ in DR-I	A decimal divide error is detected in a Divide Decimal /FD/ operation.

The following conditions can cause the I/O attention stop:

Not Ready. Not Ready conditions in card I/O devices are Hopper Empty, Stacker Full, Chip Box Full, and Cover Open. Also, the condition which exists after operation of the I/O device Stop key or when power is first turned on will cause a not ready condition.

Further Not Ready conditions on the 2560 MFCM are Primary Hopper Check or Secondary Hopper Check.

The Ready Indicator on the card I/O device is extinguished when one of these conditions exist. The Attention Indicator, also on the card I/O device, is turned on for the Stacker Full, Chip Box Full, or Cover Open conditions. The Primary Hopper Check and Secondary Hopper Check conditions on the 2560 MFCM also have individual indicators.

The Stacker Full, Chip Box Full, or Cover Open condition must be corrected by the operator before the device may be returned to a ready condition. It is not necessary to correct the Hopper empty condition in order to place the device in a ready condition.

When the card I/O device has been placed in a ready condition, the program may be continued by operating the Start key on the CPU console.

Read Check or Punch Check. All card I/O devices have one or the other of these indicators and some devices have both. The read check condition indicates that the last card read was sufficiently off register, either in punching and/or feeding, to cause inaccurate reading or that the last card read contained an invalid combination of punches in at least one column. The Punch Check condition indicates that the last card punched was punched incorrectly.

The read check and punch check conditions are normally tested and reset by the program. An I/O attention stop will occur for these conditions only when they exist and have not been tested by the program. These conditions may be reset by the I/O Check Reset key on the CPU console. The program can be continued by operating the Start key on the CPU console.

Feed Check. A feed check condition in a card I/O device indicates that a card has failed to feed properly, that a card is mispositioned in the card path, or that a malfunction has occurred. When the Feed Check indicator on the device is on, the Ready Indicator, also on the device, is off.

The feed check condition may be reset by the Non-Process Runout Key on the card I/O device only. Conditions in the printer for which I/O Attention Stops occur are:

Not Ready. Not ready conditions in the printer are End of Form, Form Check, Carriage Interlock, or Type Bar Drive Check. Also, the condition which exists after operation of the Stop key or when power is first turned on will cause a not ready condition. The Ready indicator on the printer is turned off when any one of the above conditions exist. The End of Forms, Form Check, and Carriage Interlock conditions have individual indicators on the printer.

The printer may be returned to a ready condition by correcting the condition indicated and operating the Start key on the printer. The program may then be continued by operating the Start key on the CPU console.

Print Check. The print check condition indicates that the last line printed was incorrect. The print check condition is normally tested and reset by the program. An I/O attention stop will occur for this condition only when it exists and has not been tested by the program.

The print check condition may be reset by the I/O Check Reset key on the CPU console. All other I/O operating in process when an I/O attention stop occurs will be completely executed.

Micro-Program Description (Attention Stop)

In any I/O micro-program, when an I/O attention error is detected, the micro-program continues as described in the paragraphs below and as shown in step 112 on Figure 3-27a.

112. Clear DR-R to /0/ for display purposes.
113. Move 15 (/F/) into DR-T to indicate to the customer that an I/O attention error is present (DR-T is later moved into DR-P).
114. An address exchange moves the contents of DR-S, T, and R into the sub-register to save the contents of DR-T and R.
115. The Op code address of the I/O instruction, which can not be executed because the selected I/O unit is not ready, is moved from the I Recall Addr register into DR-E, S, T, and R by the Reg to ESTR sub-routine.

116. The content of E, S, T, and R is moved into the I Addr Reg by the ESTR-to-Reg subroutine. The I Addr Reg now contains the Op code address of the I/O instruction. This enables the customer to perform this instruction by pressing the Start key.

NOTE: Before the Start key is pressed, the condition which caused the Attention stop must be corrected.

117. An address exchange moves the contents of the subregister into the DR's and the Op code address of the I/O instruction into the subregister.

118. Move the contents of DR-R (/0/) into DR-I to clear DR-R for display purposes.

119. Move the Attention stop indicator 15 from DR-T into DR-P for display purposes.

120. An address exchange moves the Op code address of the I/O instruction into 4K, S, T, and R.

63. The Op code addressed by the content of 4K, S, T, and R is read out of main storage into DR-U and L (The Op code is regenerated).

10. A control 0 stops the CPU by resetting the Process latch.

The DR's contains the following values:

Data Register	Contents
U and L	The Op code of the stopped I/O instruction.
E, S, T and R	The address of the Op code in DR-U and L.
P	/F/ to indicate an Attention stop.
I	Cleared to /0/.

POWER ON SEQUENCE

- Power is applied to the system in steps to avoid undesired side-effects.
- Steps are necessary since transistors are unstable until full power is applied.
- Power On Sequence is as follows:
  1. Turn Main Line Circuit Breaker On.
  2. Press System Reset key
  3. Press Power On key until Power On light comes on.

Description (Figure 4-1)

1. Main line circuit breaker on. This supplies the control transformer with AC current immediately. The control transformer supplies the control relays with 24V AC. The contactor K3 supplies the convenience Outlets with AC Power from the Main Line. The Power Failure Light goes on.

2. System Reset Key on. This key energizes the control relay RY3 if no overheating condition was detected by any of the thermal switches in the gates and I/O units. RY3 is held by its own hold contact and makes the Power On Key operative. It also resets the Power Failure Light (from any possible previous power failure).

3. Power On Key on. This key picks contactor K1. K1 supplies the power transformer T2 with AC current. Now all the individual DC Voltages come up. As soon as they are present - this fact is detected by the Voltage Sense Relays - Relay RY1 is picked and parallel to it Relay RR1. The Power On key actually deenergizes RY3, however, it is held by a delay capacitor long enough until a hold circuit is established for RY3.

4. Contact RY1-AL establishes a Hold Circuit for both Relays (RY1, RR1). Contact RY1-BL transfers and turns on the Power ON Light together with contactor K2. K2 supplies all I/O devices with AC power and starts the Gate Blowers. The Auxiliary contact 1 of K2 establishes a Hold Circuit for the Main Transformer Contactor K1.

5. The K2 Auxiliary 2 contact starts a delay circuit circuit on SMS Card Pos. 3 which picks Relay RR2 after 120 - 930 mili seconds.

6. Relay RR2 of Card 3 then picks Relay RR2 of SMS Card Pos. 4. This Relay sends the "Storage Protect" signal which allows the core storage timing circuits to run. Contact RR2-2 picks Relay RY2 at the same time.

7. Contact RY2-AU starts the second delay circuit on SMS Card 3 which picks Relay RR1 of SMS Card 3 after 33-85 milli seconds.

8. This Relay picks the Power Failure Relay RY5 and the Relay RY4. Relay RY4 turns on the Power On Latch and establishes another Hold Circuit for K1. This Hold Circuit is needed for the Power Off Sequence.

POWER OFF SEQUENCE

- Power is taken off the system in steps to avoid undesired side-effects.
  - It is only necessary to press the Power Off key to remove power from the system.

Detailed Description

1. Power off key is pressed. This key de-energizes RY1 and RR1. At the same time RY5 is dropped to avoid simulating a power failure condition.

2. Due to the open RY1 AL contact, RR2 on SMS Card 3 drops as soon as its delay circuit permits it.

3. RR2 drops RR2 on SMS Card 4. This drops Relay RY2.

4. Due to the open RY2-BL contact the Power ON Light and the I/O contactor K2 goes off.

5. The open RY2-AU contact then drops the RR1 Relay on SMS Card 3 as soon as its delay circuit permits it. This will take approximately 2.2 - 4 seconds.

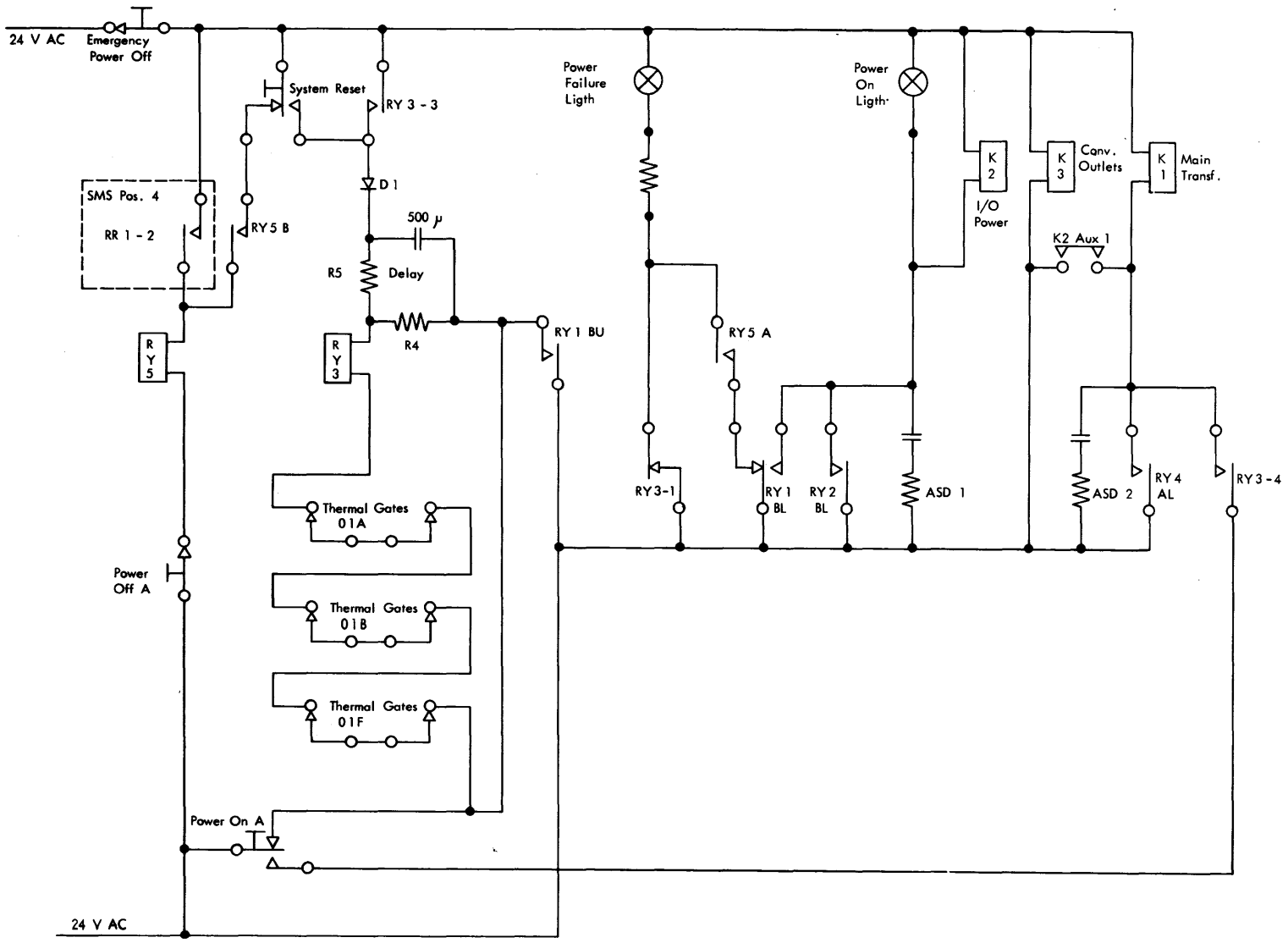


Figure 4-la Power On/Off Control

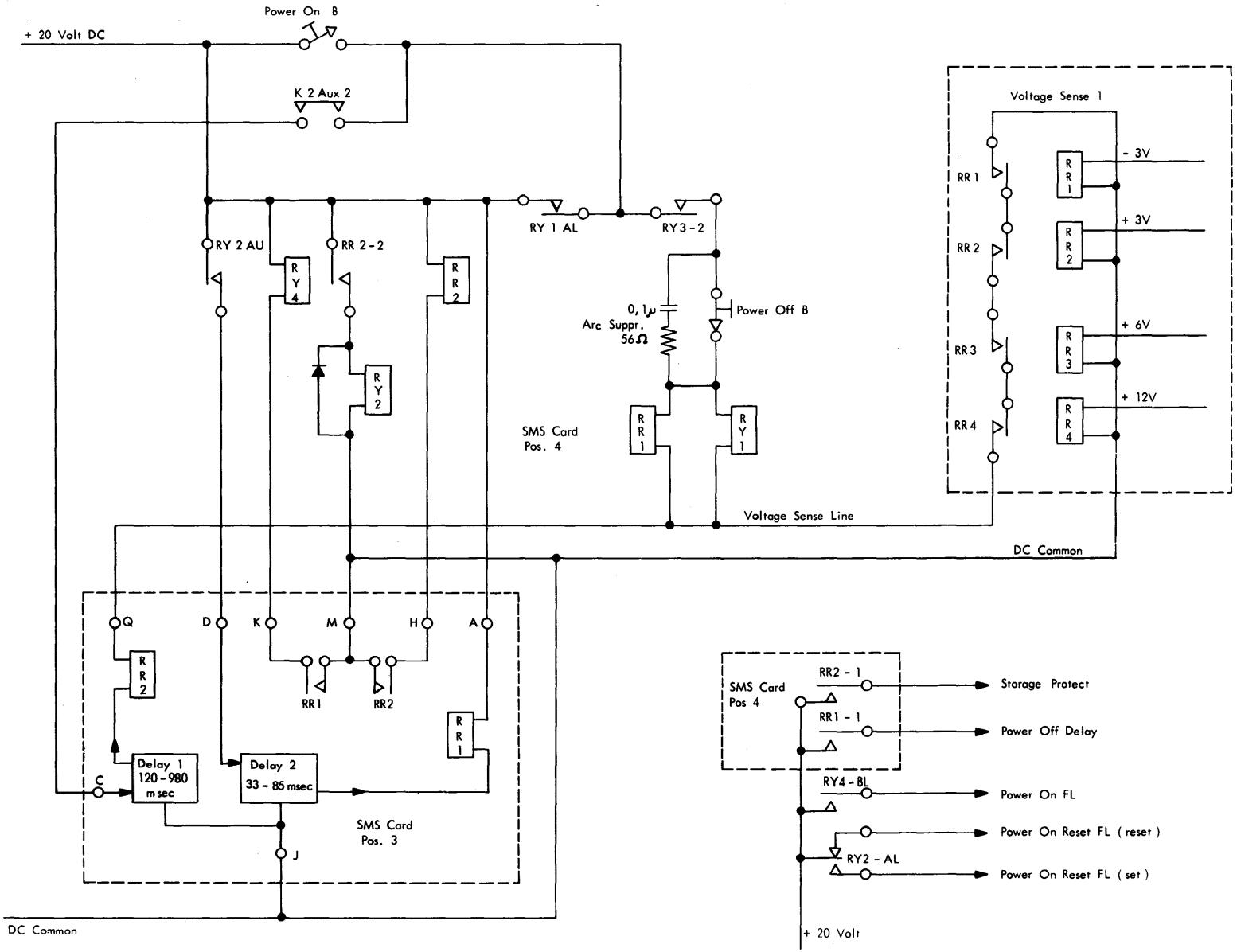


Figure 4-1b Power On/Off Control

6. When RRl drops, Relay RY4 is deenergized which causes the Main Transformer Contactor Kl to drop. Now all DC Volts are removed from the System. With RY3 still energized there is no need for pressing the System Reset Key before pressing Power On again.

Emergency Power Off. When the Emergency Switch is pulled, all control relays drop at once. After an emergency, the switch must be manually reset. Only then may the system Reset key be pressed again. This puts the supervisor relay RY3 on again and Power On may be pressed.

Power Failure

The loss of any one (or all) DC Voltage causes the Relays RYl and RRl to drop. The RYl-BL Contact turns on the Power Failure Light because RY5 is energised. The Power Failure Light stays on until all DC Voltages are raised successfully by pressing Power ON again. The only other way to extinguish the Failure Light is either pressing "Power Off" or "System Reset". In both cases Relay RY5 is dropped. However, when Power ON is attempted with the power failure still present, Relay RY3 is dropped and the Failure Light comes on again.

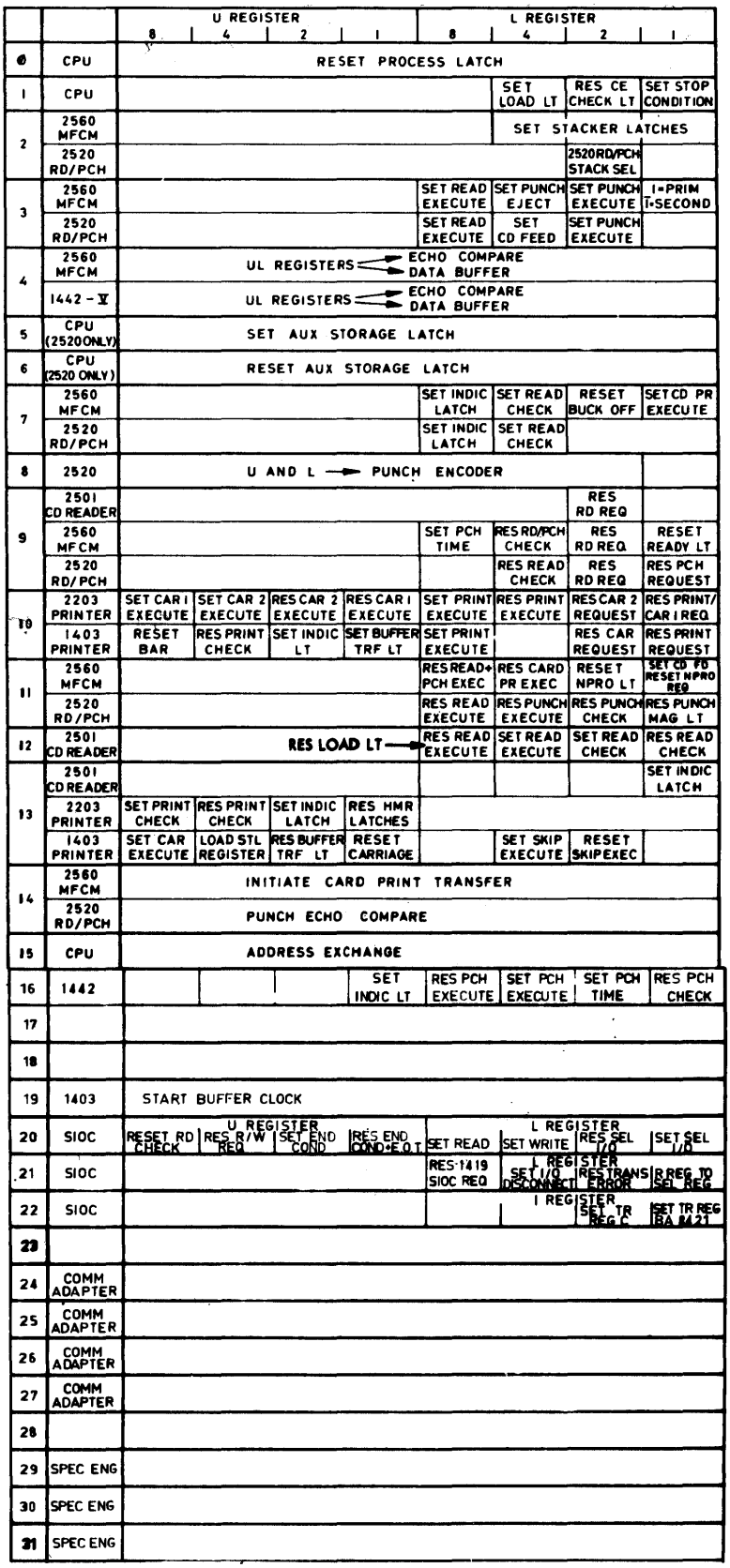


Figure 5-8 Control Instructions 0-31

5.1 CPU INDICATOR LIGHTS AND SWITCHES

(Figure 5-1)

5.1.1 Normal Lights

Power On

Power On light indicates that power has been applied to the system by pressing the Power On key.

Stop.

This indicator lights when the Process latch is off.

Register Display

The eight data registers in the CPU are displayed on the console. The functions of these register displays are explained in the sections on the Mode Switch and Operating Conditions.

5.1.2 Check Lights

Process

The CPU stops with the Process (PROC) indicator on when any of the 8 possible errors occur in the CPU. This condition may be reset only by operating the System Reset Key.

Card I/O #1

This indicator is on when a condition which requires the attention of the operator exists on Card I/O device #1, the 2501 Card Reader. Refer to the section on Operating Conditions for further details.

Card I/O #2

This indicator is on when a condition which requires the attention of the operator exists on Card I/O device #2. Refer to the section on Operating Conditions for further details.

Card I/O #3

This indicator is on when a condition which requires the attention of the Operator exists on Card I/O device #3. Refer to the section on Operating Conditions for further details.

Printer

This indicator is on when a condition which requires the attention of the operator exists on the printer. Refer to the section on Operating Conditions for further details.

5,1.3 Keys and Switches

System Reset Key.

(Description not available.)

Mode Switch

(Description not available.)

Emergency Off Switch

The Emergency off switch removes all power from the system without the usual power off sequence.

Lamp Test Switch

This switch checks for burned out indicator lamps. All indicator lamps in the system are lit with +6 v for visual checking. Indicator lamps are located:

1. On the CPU Customer Console
2. On the CPU CE Console
3. Plugable CE indicators in the CPU
4. Control panels on the connected I/O units.

I/O Check Reset Key

The I/O Check Reset Key resets all I/O Check latches (except a Feed Check) when an I/O error is present. A card feed check in any I/O unit can be reset only by operating the NPRO key in the associated I/O unit. An I/O error stops the CPU by means of the I/O Attention stop.

Power On Key

The Power On key initiates the power on sequence if power is not on. The System reset micro-routine is started as soon as power is present.

Power Off Key

The Power Off key starts the power off sequence if power is on.

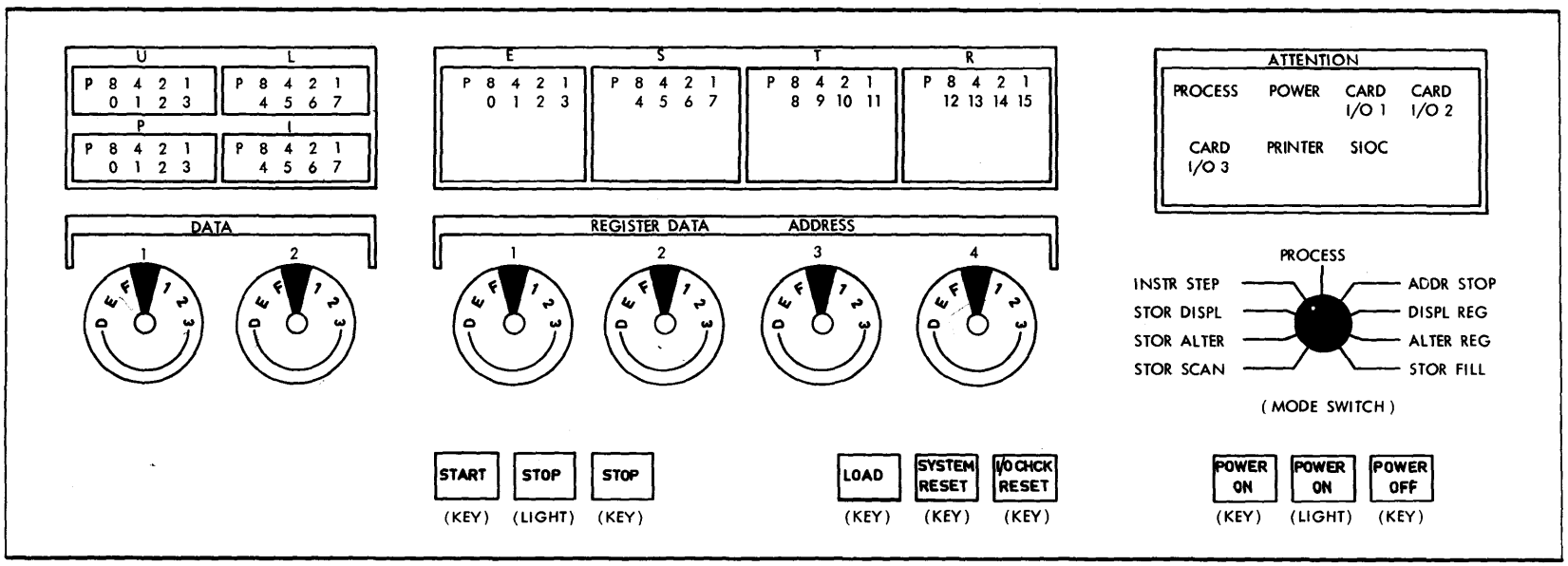


Figure 5-1 Customer Console

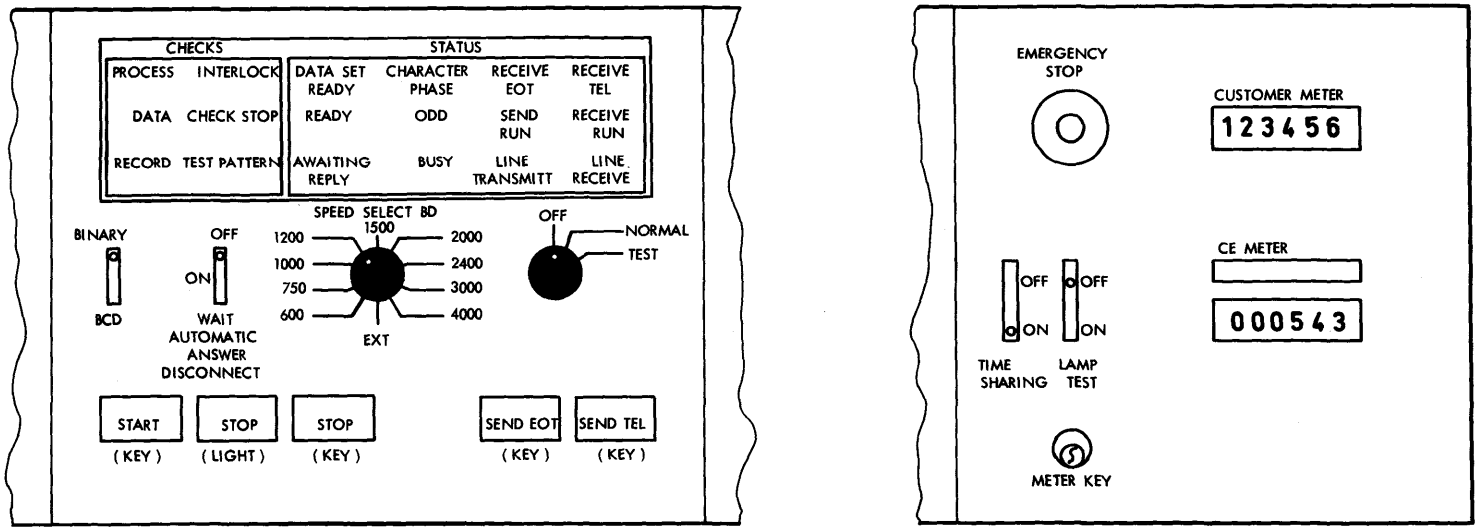


Figure 5-1a Customer Console Meter - CA Panel

Start Key

The Start Key starts the micro-program to execute the micro-program loop specified by the setting of the Mode switch. The Start Key is inoperable if a CPU error is present (Process check). The Start Key starts one of the following routines depending upon the setting of the Mode switch,

1. Process
   The CPU starts the machine program with an I phase.
2. Address Stop
   The CPU starts the machine program with an I phase.
3. Single Instruction
   The CPU starts the machine program with an I phase.
4. Storage Display
   The CPU starts the Storage Display routine.
5. Storage Alter
   The CPU starts the Storage Alter routine.
6. Register Display
   The CPU starts the Register Display routine.
7. Register Alter
   The CPU starts the Register Alter routine.
8. Storage Scan
   The CPU starts the Storage Scan routine.

Address Switches

The Address Switches 1, 2, 3, and 4 have multiple uses as follows:

1. Their status (address) is set into DR-E, S, T, and R by an Address Stop, Storage Display, or Storage Alter operation when the Mode Switch is set to one of these positions and the Start Key is pressed.
2. The Address Switches contain (are set to) the address where it is desired to stop in the machine program. A stop can occur only at a machine program Op Code address. The Mode Switch must be in the Address Stop position.
3. The Address Switches contain (are set to) the address where it is desired to stop an operation in the micro-program. The CE micro-address Stop Switch must be on.
4. Their status (data) is set into any AUX storage register /X0-X1/ (All Address and General Register). Data Switch 1 is used to select one of the 16 registers.
5. Any desired Main Storage Start Address can be generated with Address Switches 1, 2, 3 and 4.

Data Switches

Data Switches 1 and 2 have multiple uses as follows:

1. Their status (data) is set into DR-U and L by a Storage Alter operation.
2. The status (address) of Data switch 1 is set into DR-P to address the AUX-storage registers in locations /X0-X1/ by a Register Display or a Register Alter operation.

Stop Key

The Stop Key stops the program in process if the Mode switch is in the Process or Storage Scan position.

5.2 CE SWITCHES AND LIGHTS (Figure 5-2)

5.2.1 CE Switches

Prevent I/O Execute Switch

Input/output instructions are not executed when this switch is on. When an input/output instruction is encountered in the program, the central processing unit continues to the next sequential instruction.

Immediate I/O Error Stop Switch

When an input/output error occurs and this switch is on, the central processing unit stops immediately, without completing the input/output operation(s) in progress.

Process Check Override Switch

When this switch is on, the central processing unit will not stop in the event of a process check. The central processing unit will not recycle on a process check if this switch and the Recycle switch are both on.

Recycle Switch

When the Recycle switch is on, the CPU will recycle to the micro-instruction defined by the ROS Address switches when a process check condition occurs provided that the Process Check Override Switch is not on. When a ROS address stop is encountered, the CPU will recycle to the micro-instruction defined by the ROS Address Switches.

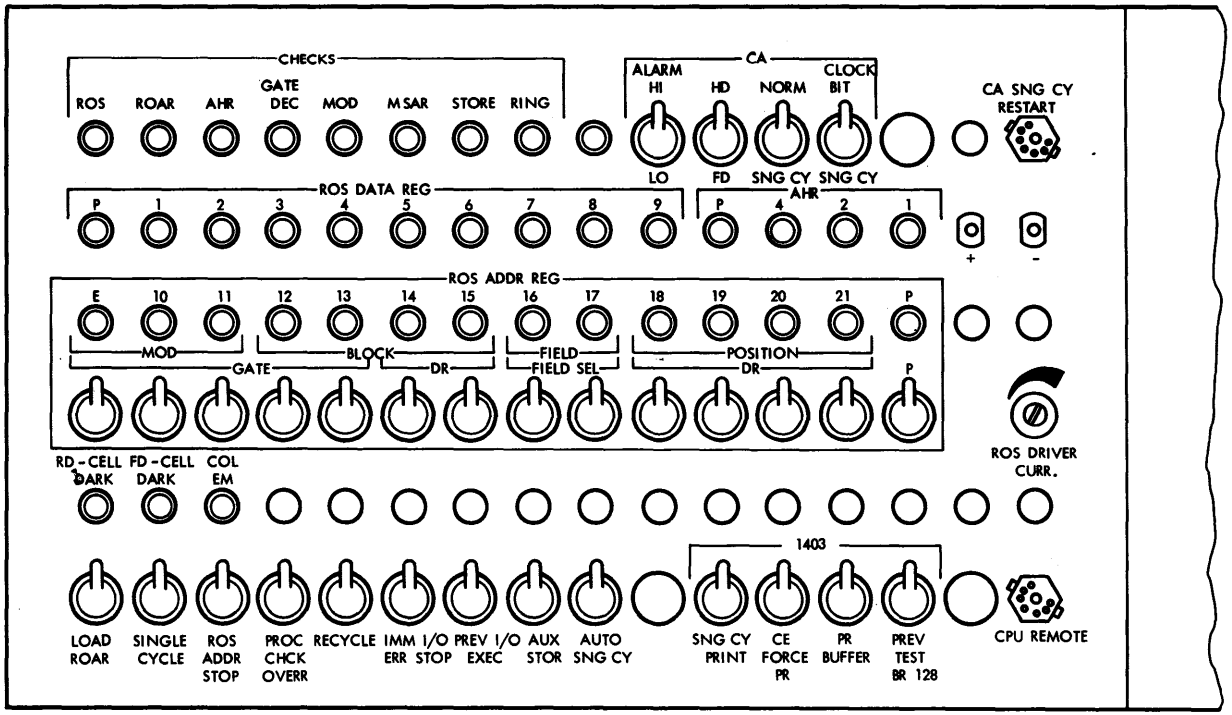


Figure 5-2 CE Console

ROS Address Stop Switch

When this switch is on, the CPU will stop when the micro-program reaches the address indicated on the Address switches (located on the CPU console) provided that the Recycle switch is not on.

If the Recycle switch is also on, the CPU will not stop when the micro-program reaches the stop address indicated on the Address Switches, but will recycle to the micro-instruction defined by the ROS address switches on the CE Panel and continue.

With both the ROS Address Stop and Recycle switches on, the CPU may be operated in a continuous loop with defined first and last ROS addresses. The ROS address of the first micro-instruction to be executed in the loop is defined by the ROS Address switches on the CE Panel. The ROS address of the last micro-instruction to be executed in the loop is defined by the Address Switches on the CPU console.

To initiate operation in the continuous loop mode from a stopped state of the CPU, the Customer Engineer must first set up the ROS addresses of the first and last micro-instructions as described above, operate the Load ROAR switch, and then operate the Start key.

The CPU will then continuously cycle between the first and last micro-instructions in the loop, provided that:

1. A process check is not encountered within the loop, or
2. the Process Check Override switch is on.

In this mode of operation, the Recycle Switch must be turned off before the CPU may be stopped.

ROS Address Switches

The fourteen ROS Address switches are located on the CE Panel in vertical correspondence with the ROAR indicators with which they are associated. These switches also define the starting address in the ROS re-cycle mode.

Load ROAR Switch

This switch is used to enter the contents of the 14 ROS address Switches on the CE Panel into the Read Only Storage Address Register.

Micro-Step Switch

The CPU will execute one micro-instruction for each operation of the Start key when the Micro-Step switch is on.

AUX Storage Switch

This switch must be turned on when the Customer Engineer desires to display or alter the contents of any one of the 256 positions of Auxiliary Storage.

NOTE: All Address Register and General Register positions can be altered or displayed by the register alter or register display routines.

AUX Storage is organized as a 16 x 16 array. Address Switch 3 is employed to address one axis of the array and Address Switch 4 is employed to address the other axis. The selected position of AUX-storage may then be displayed or altered by use of the Storage Display or Storage Alter positions of the Mode switch on the CPU console.

The storage scan mode may also be employed in the auxiliary storage area when the AUX-storage switch is on. The functions are the same as in the storage scan of main storage except that the scan is confined to Auxiliary Storage.

Auto Single Cycle

- When the Auto Single Cycle switch is turned on, the machine runs at half normal speed. (Half Cycle Mode).
- Every second Clock A and Clock C signal is suppressed.
- When an error occurs and the processor stops, the bit configuration of the instruction which caused the error is displayed on the panel.

Circuit Description (Figure 5-2a)

When the Auto Single Cycle switch is turned on, a signal is sent to the Micro Step C latch. This signal turns on the Micro Step C latch with each Pulse A. The Micro Step C latch sends a signal which satisfies an AND switch of the Delta Process latch. This AND switch turns off the Delta Process latch. When the Delta Process latch is off, the Clock A signal goes off also. At the same time the Clock C signal goes off.

Since the TROS Timing depends upon Clock A and Clock C (ROAR), an instruction read out of the TROS is present as long as no new Clock A signal reads out a new one. Due to the suppression of every second Clock A and Clock C cycle the erroneous instruction is displayed when the CPU stops on an error.

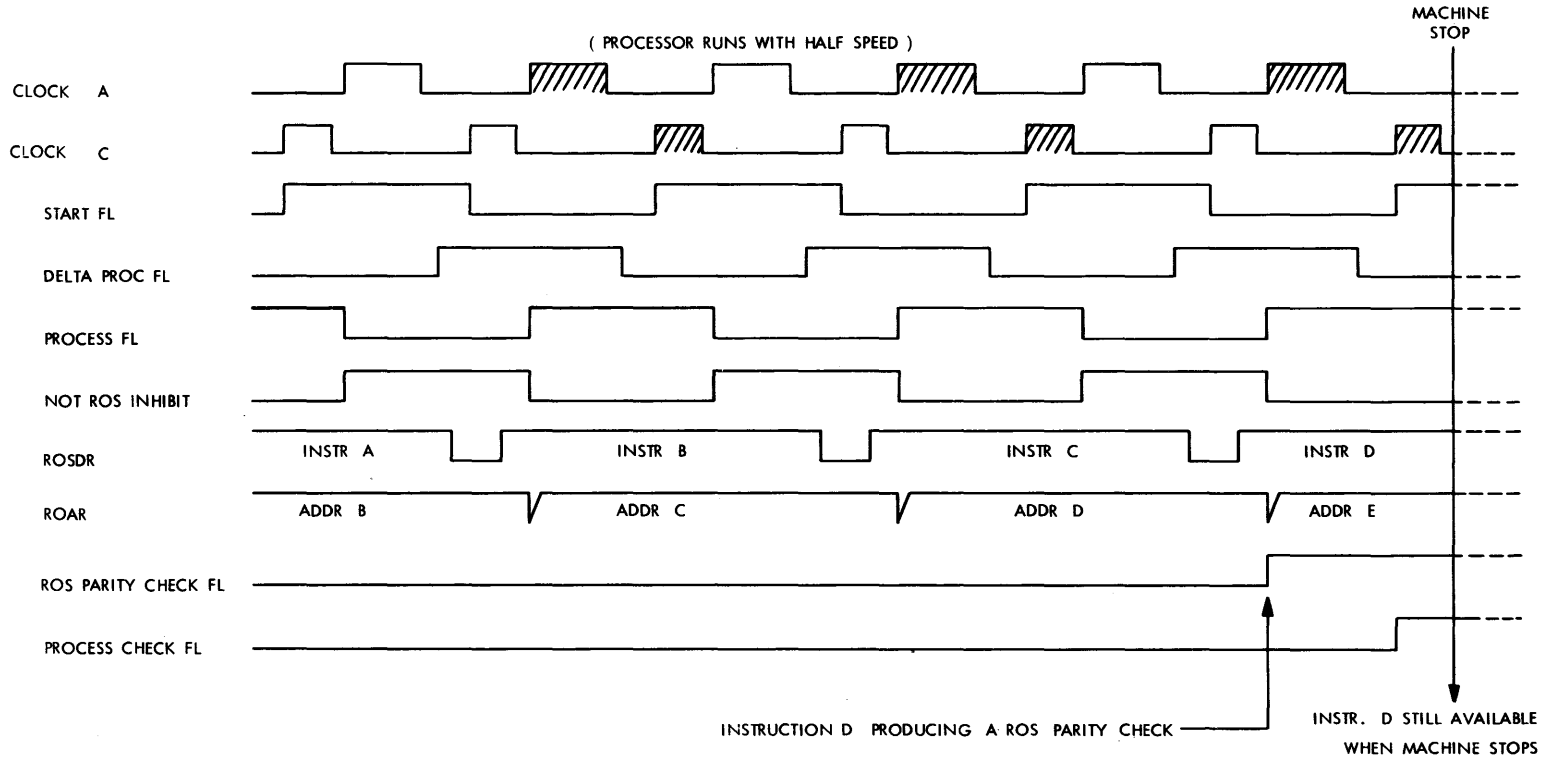


Figure 5-2a Automatic Single Cycle Mode Sequence

Remote Control Unit

A receptacle is provided on the CE Panel for a Remote Control Unit. The Remote Control Unit contains a Start key, Stop key, and a System Reset key.

The Start key on the CPU console is disabled when the Remote Control Unit is plugged in. The Stop and System Reset keys in the Remote Control Unit are in parallel with the respective Start and System keys on the CPU console.

5.2.2 CE Indicator Lights

ROAR Indicators

The thirteen bits of the Read Only Storage Address Register (ROAR) and the On status of the Even ROS Address latch are displayed in these CE Panel indicators.

ROSDR Indicators

The nine operation code bits of the Read Only Storage Data Register (ROSDR) and the parity bit for the entire micro-instruction are displayed in these CE Panel indicators.

Check Indicators

The central processing unit check conditions which are indicated on the CPU console by the Process Indicator are displayed individually in these CE Panel indicators. These conditions are:

ROS Check	STAR Check
Ring Check	ROAR Check
Modifier Check	Gate Decoder Check
Store Check	AHR Check

AHR Indicators

The four bits (3 bits plus parity) of the AHR are displayed in these CE Panel indicators.

Feed Check Indicators

Three of the card jam or malfunction conditions which cause a feed check in the card I/O devices are indicated individually on the CE Panel. These conditions are:

1. Column Emitter Check
2. Read Solar Cell Check
3. Feed Solar Cell Check

All card I/O devices included as system components are connected to these check circuits. The particular device in which one of these conditions occur may be determined from the Feed Check indicators on the card I/O devices.

5.3 SUMMARY CHARTS

This section consists of figures which summarize significant machine elements as follows:

Figure 5-3 Code Tables
Figure 5-4 Micro-instruction Summary
Figure 5-5 Op Code Table
Figure 5-6 Auxiliary Storage Layout
Figure 5-7a Sense Instructions 0-31
Figure 5-7b Sense Instructions 32-63
Figure 5-8 Control Instructions 0-31
Figure 5-8a Hexadecimal-decimal Conversion Table

5.4 SUPPLEMENTARY DIAGRAMS

5.4.1 How to read the Simplified Logic Diagram SLD

Title Block

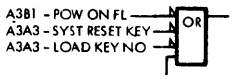
Title. Each page of the SLD has a heading in the upper right hand corner (Figure 5-9).

The title consists of a number which tells to which machine the diagram belongs, for example, 2020 is the System /360 Model 20 processor. One or more descriptive names indicate what parts of the machine are shown on the page, for example, Bus and the Modifier.

Logic Number. Each page also has a Logic Number, C3 in this example. This Logic Number is found on any number of signals throughout the entire SLD. It indicates from which Logic page the signal comes. In this manner it is possible to trace a signal back to its source.

Cross References

To further pin point a signal on the Logic page each page is divided by fields. These fields are defined by vertical and horizontal coordinates. Horizontally the fields are labelled A, B, C, D, from left to right and vertically 1, 2, 3, 4 from top to bottom. Each signal also has this information besides the Logic Number.



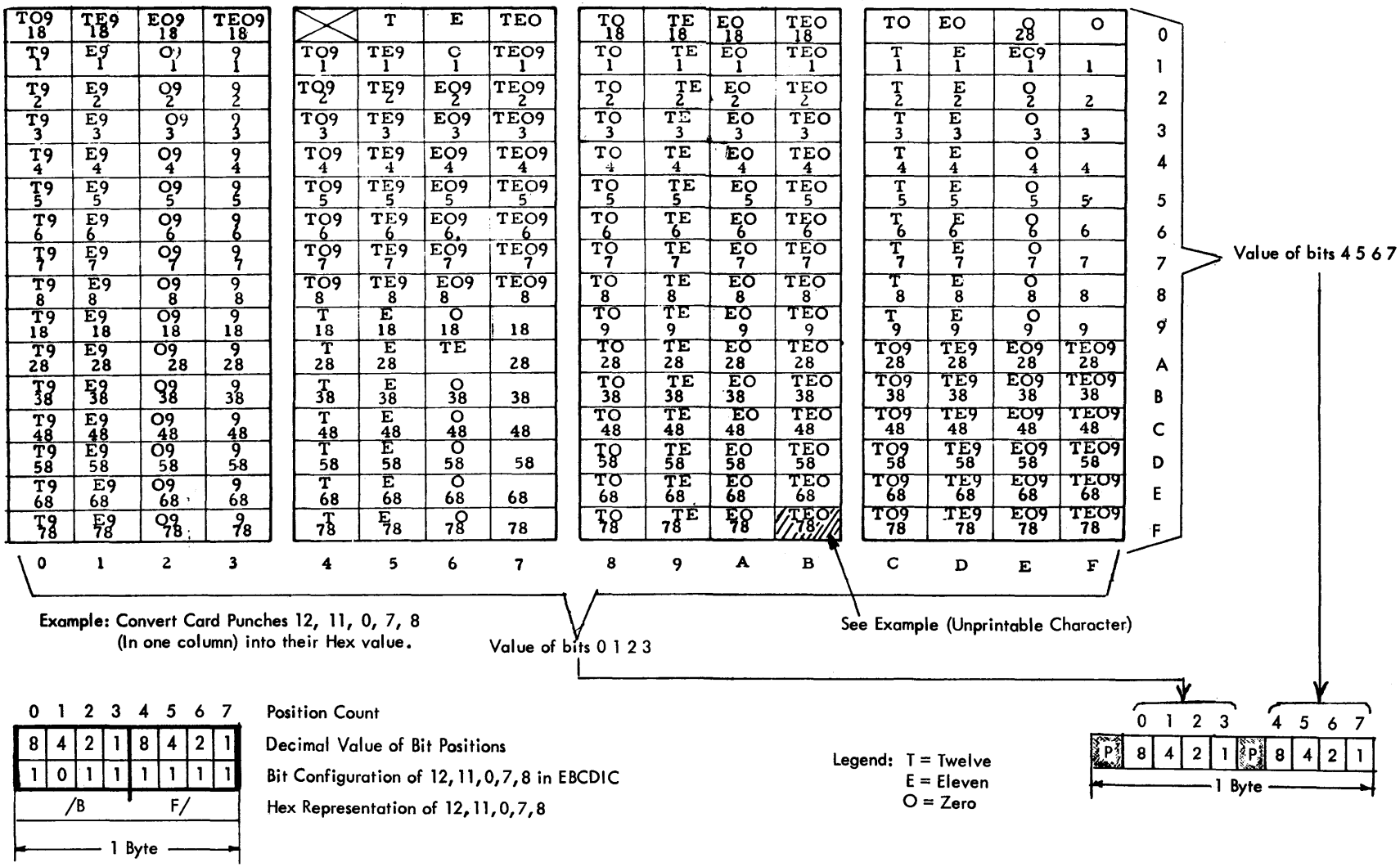


Figure 5-3a Conversion of EBCDIC Card Code to Hexadecimal and from Hexadecimal into EBCDIC Bit Configuration

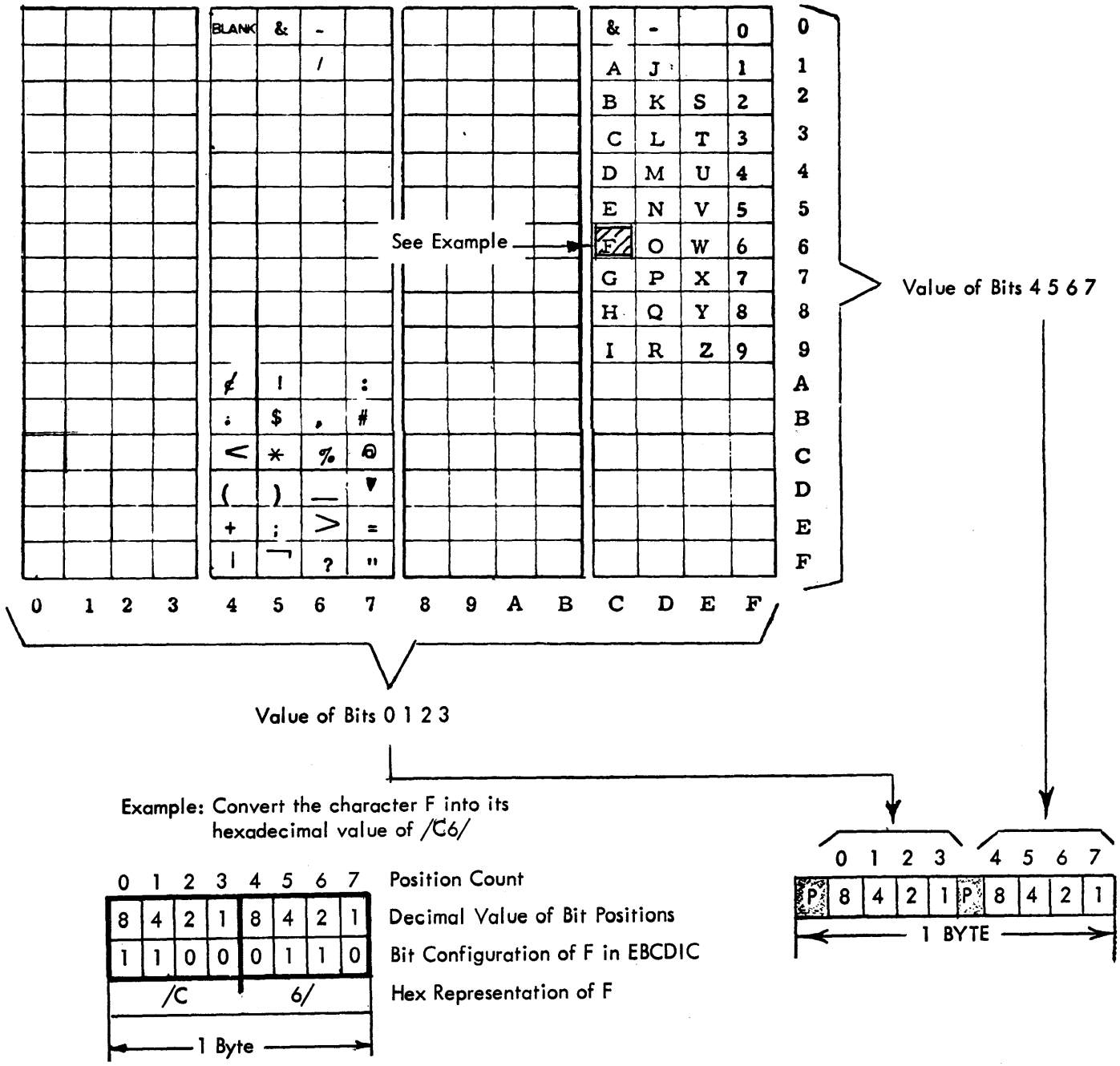


Figure 5-3b Conversion of Characters into Hexadecimal Code and from Hexadecimal into EBCDIC Bit Configuration

SLDA REPR	C A S REPR.	ROSDR Bits 0	1	2	3	4	5	6	7	8	9	10 - - - - - - - 17 18 - - 21	
MX*X	X←X	P	0	1	X*	X*	X*	0	X	X	X	← NSI Address →	Transmit the content of a selected data register (X) into another selected data register (X*)
MX*N	N←X	P	1	1	X*	X*	X*	N	N	N	N		Transmit the "N" part of the micro instruction, a binary value between 0 and 15, into a selected data register (X*)
UOX	UOX	P	0	0	0	1	1	1	X	X	X	Mask	Use the content of a selected data register (X) and OR it with the NSI address to determine if a branch in the microprogram is required
UAX	UAX	P	0	0	0	1	0	1	X	X	X	Mask	Use the content of a selected data register (X) and AND it with the NSI address to determine if a branch in the microprogram is required
UXX	UXX	P	0	0	0	0	1	1	X	X	X	Mask	Use the content of a selected data register (X) and EXCLUSIVE OR it with the NSI address to determine if a branch in the microprogram is required
INCR X	X + 1	P	0	0	0	1	0	0	X	X	X		Increment the selected data register (X) by 1
DECR X	X - 1	P	0	0	0	0	1	0	X	X	X		Decrement the selected data register (X) by 1
F N	FN	P	1	0	0	1	Y	N	N	N	N		Read one Byte out of AUX storage into the U and L data register. The AUX storage ist addressed by the "N" part of the microinstruction
F PN	FPN	P	1	0	1	1	Y	N	N	N	N		Read one Byte out of AUX storage into the U and L data register. The AUX storage is addressed by the content of data register P and the "N" part of the microinstruction
F STR	FSTR	P	0	0	1	1	Y	0	0	1	1		Read one Byte out of Main storage into U and L data register. The Main storage is addressed by the content of the data register S, T, and R
S N	SN	P	1	0	0	0	Y	N	N	N	N		Write one Byte into AUX storage from U and L data register. The AUX storage is addressed by the "N" part of the micro instruction
S PN	SPN	P	1	0	1	0	Y	N	N	N	N		Write one Byte into AUX storage from U and L data register. The AUX storage is addressed by the content of data register P and the N part of the micro instruction
S STR	SSTR	P	0	0	1	0	Y	0	0	1	1		Write one Byte into Main storage from U and L data register. The Main storage is addressed by the content of the data registers S, T, and R
SENSE	SNS n	P	0	1	n	n	n	1	n	n	n		Sense the condition of various CPU and I/O Indicators and set the sensed condition into data register L.
CTL	CTL n	P	0	0	1	n	n	1	n	n	n		Control any CPU or I/O operations such as stop the CPU or release the Reader clutch
PAT	PAT	P	Z	Z	Z	Z	Z	Z	Z	Z	Z	← NSI Address →	
	PAT	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z - - - - - - - - - - - Z	

X 0-7				X* 4	X* 2	X* 1		X 4	X 2	X 1
N 0-15							N 8	N 4	N 2	N 1
n 0-63				n 32	n 16	n 8		n 4	n 2	n 1
						Y 0-1	0=Long Cycle 1=Short Cycle			
	P 0-1	Parity Bit for Address (13 Bits) and Micro-instruction (22 Bits)								
	Z	Bit Pattern for CDPR or Test								

Figure 5-4 Micro-instruction Summary

SYSTEM 360/MOD 20 INSTRUCTIONS

FIXED-POINT

NAME	MN	OP	FO
ADD	AR	1A	RR
SUBTRACT	SR	1B	RR
STORE HW	STH	40	RX
LOAD HW	LH	48	RX
COMP HW	CH	49	RX
ADD HW	AH	4A	RX
SUBTRACT HW	SH	4B	RX
*LOAD COMPL	LCR	13	RR
*LOAD	LR	18	RR
*COMPARE	CR	19	RR
MULTIPLY HW	MH	4C	RX

DECIMAL

NAME	MN	OP	FO
MOVE OFFSET	MVO	F1	SS
PACK	PACK	F2	SS
UNPACK	UNPK	F3	SS
ZERO ADD	ZAP	F8	SS
COMP DECIMAL	CP	F9	SS
ADD DECIMAL	AP	FA	SS
SUBTR DECIMAL	SP	FB	SS
MULTIPLY DEC	MP	FC	SS
DIVIDE DEC	DP	FD	SS

LOGICAL

NAME	MN	OP	FO
MOVE	MVI	92	SI
MOVE	MVC	D2	SS
MOVE NUM	MVN	D1	SS
MOVE ZONE	MVZ	D3	SS
COMP LOG	CLI	95	SI
COMP LOG	CLC	D5	SS
EDIT	ED	DE	SS
AND	NI	94	SI
OR	OI	96	SI
TEST MASK	TM	91	SI
HALT & PROC	HPR	99	SI
TRANSLATE	TR	DC	SS
*LOAD ADDR	LA	41	RX
*STORE CHAR	STC	42	RX
*INSERT CHAR	IC	43	RX

BRANCHING

NAME	MN	OP	FO
BR COND	BCR	07	RR
BR COND	BC	47	RX
BR&STORE	BASR	OD	RR
BR&STORE	BAS	4D	RX
SET PSW	SPSW	81	SI
BR ON COUNT	BCT	46	RX

INPUT/OUTPUT

NAME	MN	OP	FO
TEST I/O & BR	TIOB	9A	SI
CONTROL I/O	CIO	9B	SI
TRANSFER I/O	XIO	D0	SS

2501 CARD READER

FUNCTION	MN	OP	DA	FS
READ CARD	XIO	D0	1	2
READ COL BIN	XIO	D0	1	10
TEST READ BUSY	TIOB	9A	1	0
TEST READ ERROR	TIOB	9A	1	1
TEST LAST CARD	TIOB	9A	1	4

2560 MFCM

FUNCTION	MN	OP	DA	FS
READ PRIM	XIO	D0	2	2
READ PRIM Col Bin	XIO	D0	2	10
READ SEC	XIO	D0	2	3
READ SEC Col Bin	XIO	D0	2	11
PUNCH PRIM	XIO	D0	2	4
PUNCH SEC	XIO	D0	2	5
PUNCH&FEED PRI	XIO	D0	2	6
PUNCH&FEED SEC	XIO	D0	2	7
*WRITE CARD	XIO	D0	2	0
TEST Rd/Pch Busy	TIOB	9A	2	0
TEST Rd/Pch Error	TIOB	9A	2	1
TEST Card Pr Busy	TIOB	9A	2	2
TEST LAST CARD	TIOB	9A	2	4
TEST FEED ERROR	TIOB	9A	2	5
PRIM STACK SEL	CIO	9B	2	0
SEC STACK SEL	CIO	9B	2	1
PCH STACK SEL	CIO	9B	2	2
PRT HEAD SEL	CIO	9B	2	3

2520 READ/PUNCH

FUNCTION	MN	OP	DA	FS
READ	XIO	D0	2	2
READ Col Bin	XIO	D0	2	10
PUNCH	XIO	D0	2	4
PUNCH&FEED	XIO	D0	2	6
TEST READ BUSY	TIOB	9A	2	0
TEST READ ERROR	TIOB	9A	2	1
TEST PCH BUSY	TIOB	9A	2	2
TEST PCH ERROR	TIOB	9A	2	3
TEST LAST CARD	TIOB	9A	2	4
TEST FEED ERROR	TIOB	9A	2	5
STACK SELECT	CIO	9B	2	0

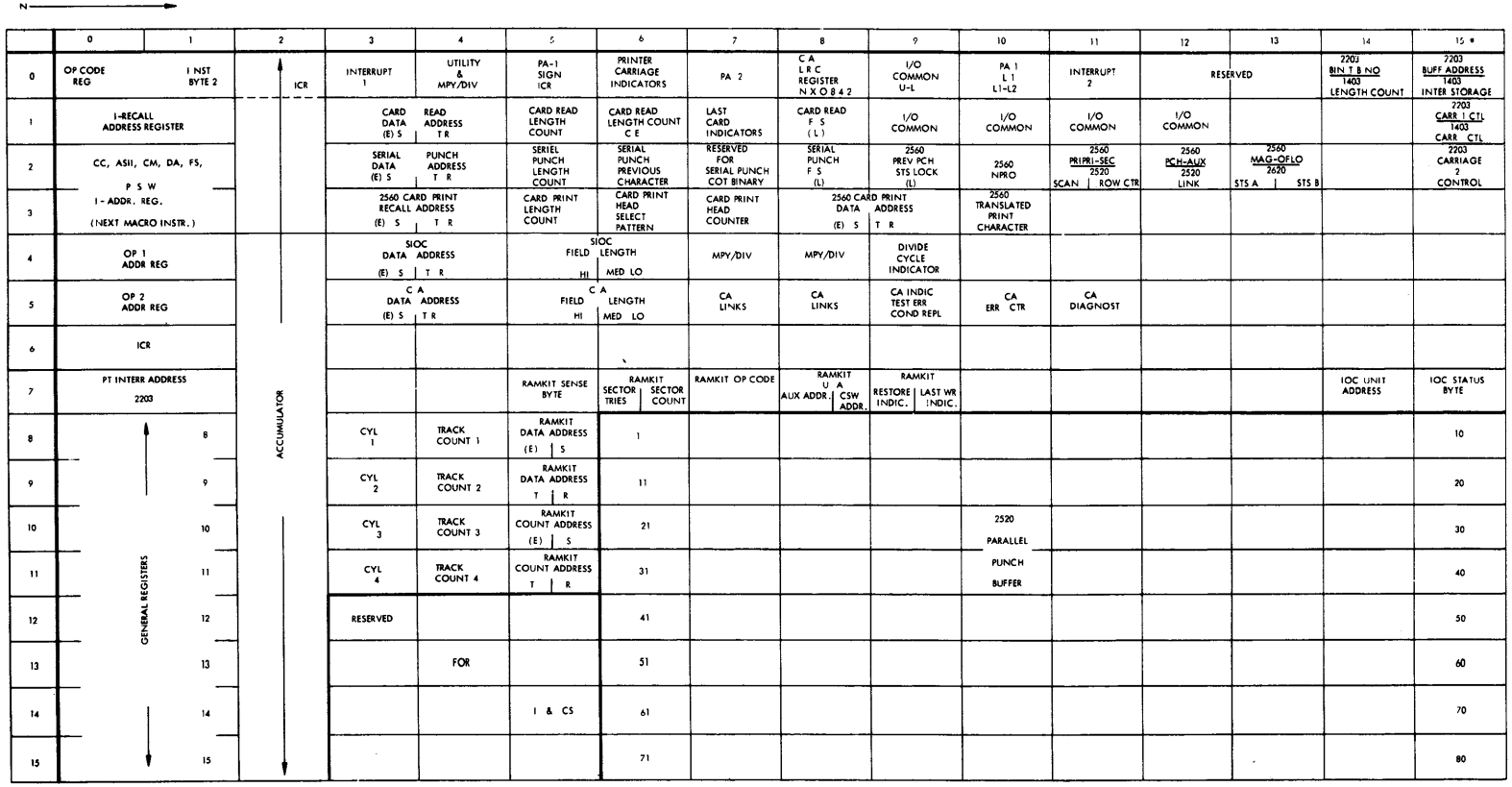
1442-5 CARD PUNCH

FUNCTION	MN	OP	DA	FS
PUNCH	XIO	D0	3	4
TEST PCH BUSY	TIOB	9A	3	2
TEST PCH ERROR	TIOB	9A	3	3
STACK SELECT	CIO	9B	3	0

PRINTER

FUNCTION	MN	OP	DA	FS
PRINT	XIO	D0	4	0
PRINT&Space Supp	XIO	D0	4	1
TEST PRINT BUSY	TIOB	9A	4	0
TEST PRINT ERROR	TIOB	9A	4	1
TEST CH 9Lower	TIOB	9A	4	2
TEST CH12Lower	TIOB	9A	4	3
*TEST CH 9 Upper	TIOB	9A	4	4
*TEST CH12Upper	TIOB	9A	4	5
IMM SPACE	CIO	9B	4	4
IMM SKIP	CIO	9B	4	5
DELAY SPACE	CIO	9B	4	6
DELAY SKIP	CIO	9B	4	7
*IMM SPACE Upp	CIO	9B	4	8
*IMM SKIP Upp	CIO	9B	4	9
*DEL SPACE Upp	CIO	9B	4	10
*DEL SKIP Upp	CIO	9B	4	11
*IMM SPACE Both	CIO	9B	4	12
*IMM SKIP Both	CIO	9B	4	13
*DEL SPACE Both	CIO	9B	4	14
*DEL SKIP Both	CIO	9B	4	15

Figure 5-5 Op Code Table



P \ N	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 *
0	OP CODE REG	INST BYTE 2	ICR	INTERRUPT 1	UTILITY & MPY/DIV	PA-1 SIGN ICR	PRINTER CARRIAGE INDICATORS	PA 2	C A L R C REGISTER N X 0 8 4 2	I/O COMMON U-L	PA 1 L 1 L1-L2	INTERRUPT 2	RESERVED		2203 BIN T B NO 1403 LENGTH COUNT	2203 BUFF ADDRESS 1403 INTER STORAGE
1	I-RECALL ADDRESS REGISTER			CARD DATA (E) S | READ ADDRESS T R		CARD READ LENGTH COUNT	CARD READ LENGTH COUNT C E	LAST CARD INDICATORS	CARD READ F S (L)	I/O COMMON	I/O COMMON	I/O COMMON	I/O COMMON			2203 CARR I CTL 1403 CARR CTL
2	CC, ASII, CM, DA, FS, P S W			SERIAL DATA (E) S | PUNCH ADDRESS T R		SERIEL PUNCH LENGTH COUNT	SERIAL PUNCH PREVIOUS CHARACTER	RESERVED FOR SERIAL PUNCH COT BINARY	SERIAL PUNCH F S (L)	2560 PREV PCH STS LOCK (L)	2560 NPRO	2560 PRIPRI-SEC 2520 SCAN | ROW CTR	2560 PCH-AUX 2520 LINK	2560 MAG-OFLO 2620 STS A | STS B		2203 CARRIAGE 2 CONTROL
3	I - ADDR. REG. (NEXT MACRO INSTR.)			2560 CARD PRINT RECALL ADDRESS (E) S | T R		CARD PRINT LENGTH COUNT	CARD PRINT HEAD SELECT PATTERN	CARD PRINT HEAD COUNTER	2560 CARD PRINT DATA ADDRESS (E) S | T R		2560 TRANSLATED PRINT CHARACTER					
4	OP 1 ADDR REG		ACCUMULATOR	SIOC DATA ADDRESS (E) S | T R		SIOC FIELD LENGTH HI | MED LO		MPY/DIV	MPY/DIV	DIVIDE CYCLE INDICATOR						
5	OP 2 ADDR REG			C A DATA ADDRESS (E) S | T R		C A FIELD LENGTH HI | MED LO		CA LINKS	CA LINKS	CA INDIC TEST ERR COND REPL	CA ERR CTR	CA DIAGNOST				
6	ICR															
7	PT INTERR ADDRESS 2203					RAMKIT SENSE BYTE	RAMKIT SECTOR TRIES | SECTOR COUNT	RAMKIT OP CODE	RAMKIT U A AUX ADDR. | CSW ADDR.	RAMKIT RESTORE INDIC. | LAST WR INDIC.					IOC UNIT ADDRESS	IOC STATUS BYTE
8	8			CYL 1	TRACK COUNT 1	RAMKIT DATA ADDRESS (E) | S	1									10
9	9			CYL 2	TRACK COUNT 2	RAMKIT DATA ADDRESS T | R	11									20
10	10			CYL 3	TRACK COUNT 3	RAMKIT COUNT ADDRESS (E) | S	21				2520 PARALLEL					30
11	11			CYL 4	TRACK COUNT 4	RAMKIT COUNT ADDRESS T | R	31				PUNCH BUFFER					40
12	12			RESERVED			41									50
13	13				FOR		51									60
14	14					I & CS	61									70
15	15						71									80

GENERAL REGISTERS

Figure 5-6 Auxiliary Storage Layout

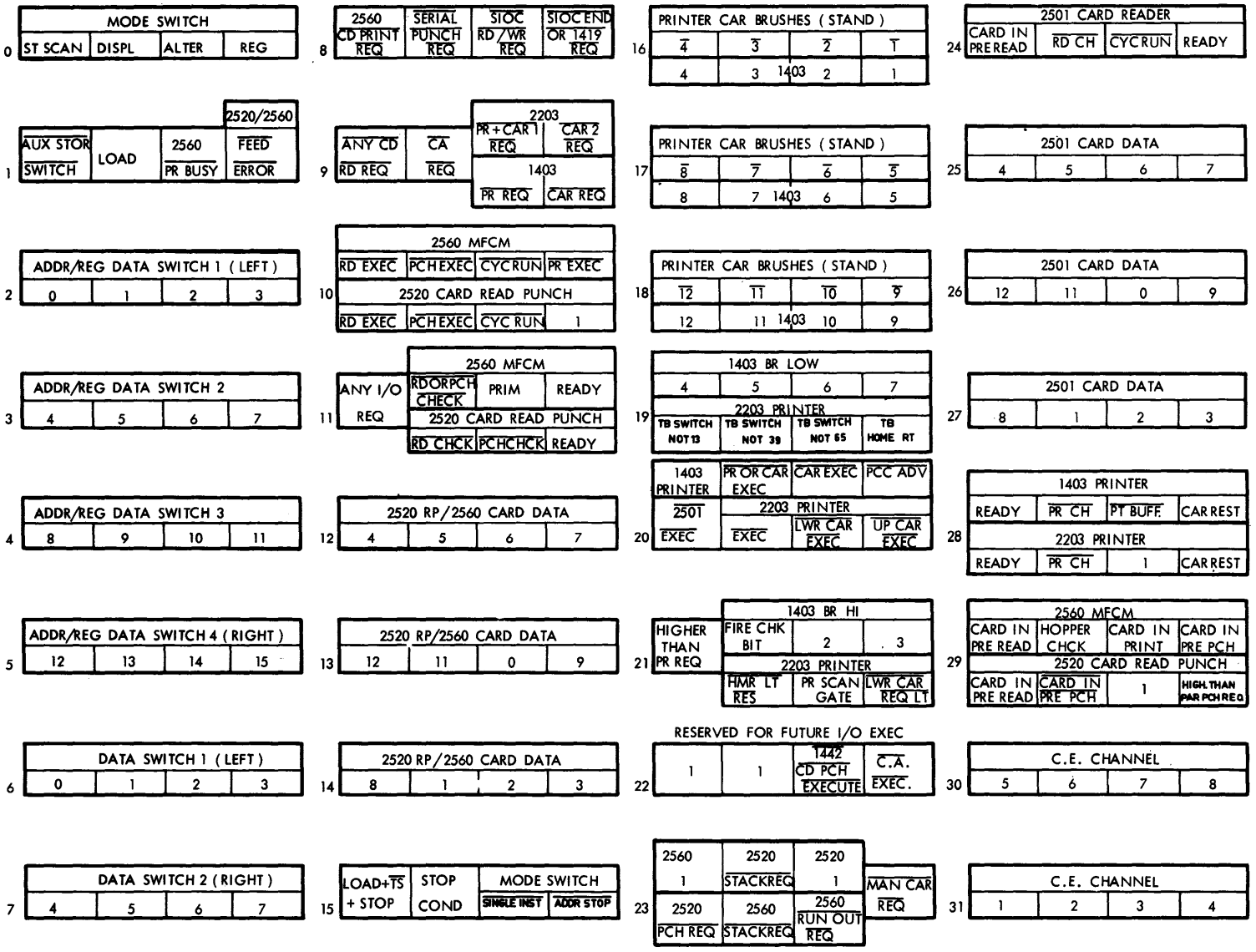


Figure 5-7a Sense Instructions 0-31

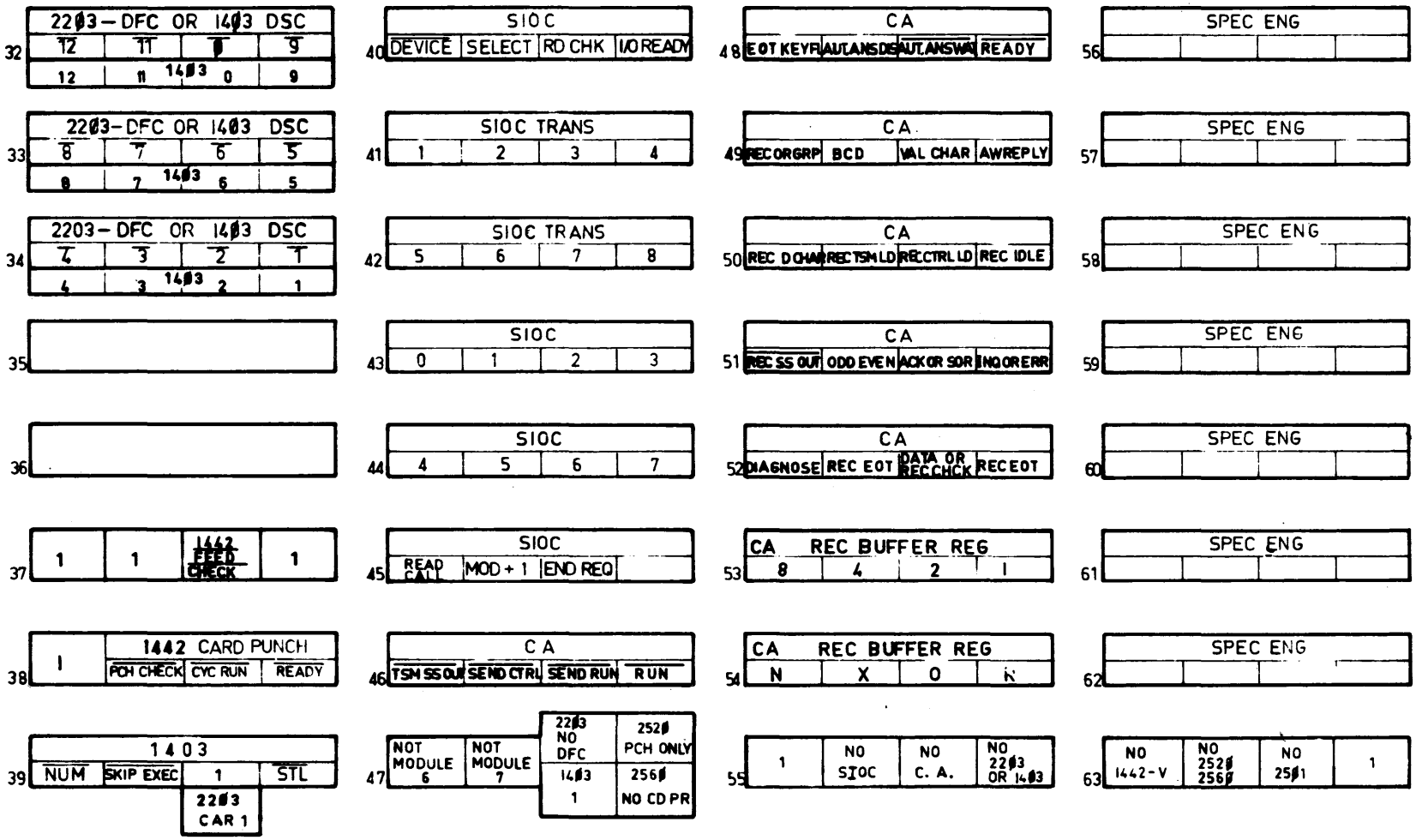
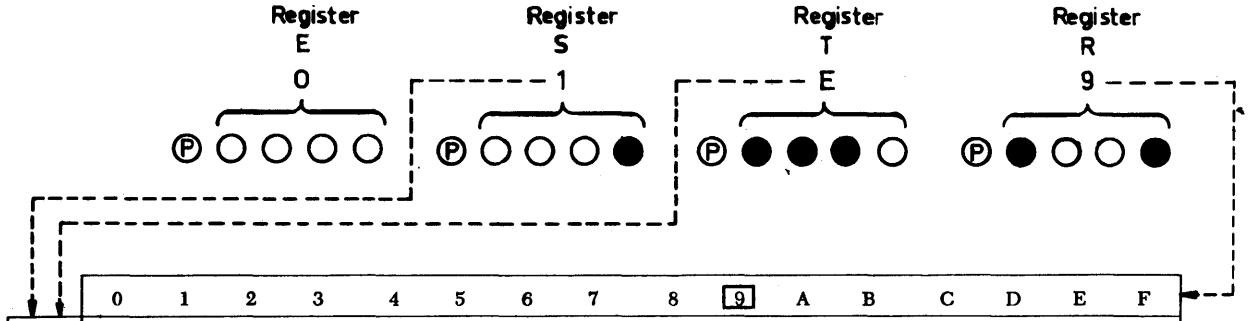


Figure 5-7b Sense Instructions 32- 63

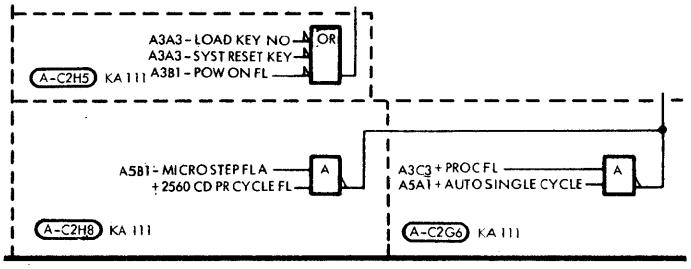
This figure illustrates how to convert hexadecimal numbers to decimal numbers. See the System/360 Model 20 FE Maintenance Manual for the complete table.



		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
0	1	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
0	2	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
0	3	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
0	4	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
0	5	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
0	6	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
0	7	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
0	8	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
0	9	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0	A	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0	B	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0	C	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0	D	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0	E	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0	F	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255
1	0	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
1	1	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
1	2	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
1	3	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
1	4	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
1	5	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
1	6	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
1	7	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
1	8	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
1	9	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1	A	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1	B	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1	C	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1	D	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1	E	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1	F	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511

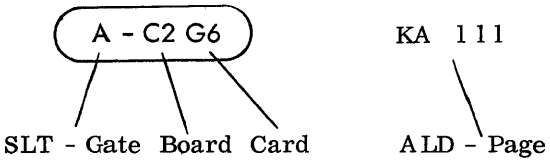
Figure 5-8a Hexadecimal - Decimal Conversion Table

In the example, an OR switch is shown with 3 input signals. All three of these signals come from Logic page Number A3. The upper most signal comes from field Bl of page A3, the other two signals come from field A3 of page A3. Field A3 means the left most section of the page and the third field from the top.



The Simplified Logic Diagram contains dotted lines which form another type of field. Each dotted field represents a card and all the hardware components which are shown within the dotted lines are actually located on such a card.

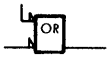
In each dotted field, a Number with an oval frame specifies the Gate, Board, and Card from left to right:



The number to the right of each card location number indicates the page of the Automated Logic Diagram on which the component is found. In this manner, direct reference to the physical layout of the machine as well as to the original wiring diagram (ALD) is made.

Basic Philosophy of the SLD Logic

AND, OR Switches

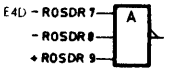


The inputs to any type of switch are always shown with the polarity required to satisfy the switch. The output of any type of switch is always shown with the polarity which it has when the switch is satisfied. In the example it is necessary to have either one of the two minus signals (indicated by the wedges) to satisfy the OR switch.

When this OR switch is satisfied, its output is plus (indicated by the absence of a wedge), thus it is an inverting type OR switch.

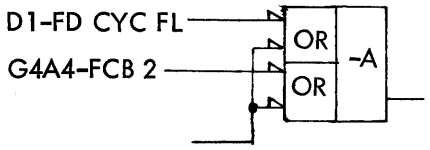
Here an example of a none inverting OR switch. Any one of the 3 input signals must be minus to satisfy the switch. When it is satisfied, the output is minus. When it is not satisfied the output is plus.

The same rules apply to AND switches. The style of drawing indicates that both input signals must be plus to satisfy the switch. When it is satisfied, the output is minus.



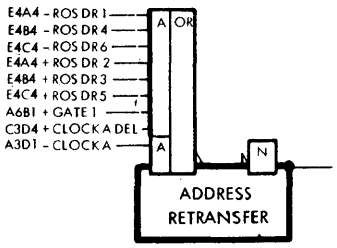
In this example the AND switch needs all three input signals to be satisfied. These inputs must all be plus. However, the incomming signals are minus ROSDR 7, minus ROSDR 8 and plus ROSDR 9. The polarity signs of the signals show the polarity of each signal when it is present. This means that in order to satisfy this switch the signals - ROSDR 7 and - ROSDR 8 must not be present but the + ROSDR 9 signal must be present.

This AND switch needs 3 minus inputs to be satisfied. The two lower signals, however, are plus when they are present. Thus, the lower signals, that is + ROSDR 1 and + Aux Stor CTRL FL must not be present to satisfy the switch. When a minus signal is not present the line is plus. When a plus signal is not present the line is minus.



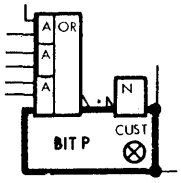
An OR - AND switch combination such as this is occasionally found in the SLD. Two OR switches which need at least one minus signal each to be satisfied consitute the two legs of an AND switch. When both legs (OR switches) of the AND switch are satisfied the output of the AND is plus.

Latches

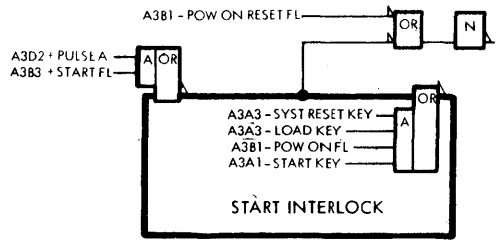


The component which occurs most often in the SLD is the Flip Latch (FL). It consists of two or more AND switches in conjunction with an OR switch.

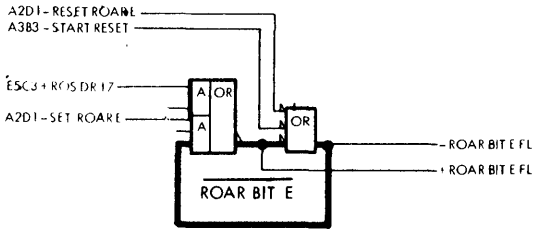
The signals required to set (turn on) this latch are marked by the heavy line where these signals enter the latch. In the example, the entire length of the upper AND switch is marked by the heavy line. Thus, all 8 signals must be present to set the latch. Satisfying the upper AND switch also satisfies the OR switch. The minus output of the OR switchis inverted by the "N" block and fed back into the latch through the lower AND switch. This is the "latch-back". The latch-back, however, is not enough to keep the latch turned on because it is only one of the legs of the lower AND switch (Latch Back AND switch). The other leg has a negative signal (when it is present) the switch however demands a plus signal to be satisfied. So the latch stays on only when the no Clock A signal is present. As a rule when the latch back conditions are not satisfied the latch is turned off.



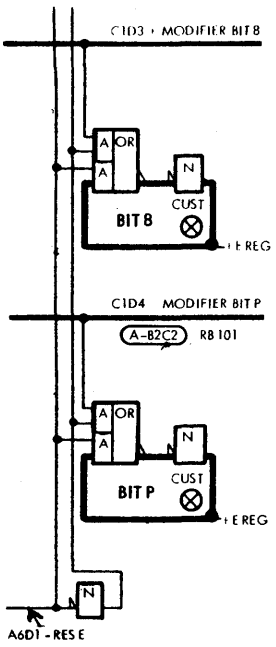
Here a latch with 3 AND switches. The set conditions can be readily seen from the heavy line. Either one of the upper two AND switches can be used to set the latch. To keep it turned on, both legs (besides the latch back) on the lower AND switch must be plus. To turn the latch off, one (or more) of the legs of the latch back AND switch must go minus.



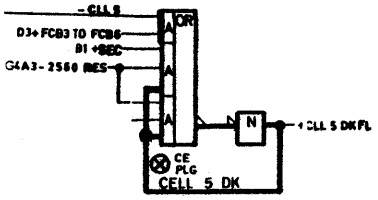
Here is a latch with a somewhat different configuration. This latch is set by the signals + Pulse A and + Start Flip Latch (heavy line). The output of the OR switch is minus and it enters another OR switch. This OR switch seems odd because it demands a plus input to be satisfied and produces a minus signal when satisfied. This is correct. The drawing indicates that this OR switch is actually a general inverter. When the OR switch receives a minus signal (as is the case when the latch is turned on) it produces a plus signal at its output. This plus signal is latched back into the first OR switch and the latch stays on. This latch can be turned off only when the 4 - legged AND switch is satisfied. When the 4 - leg AND switch is satisfied by 4 plus signals, a plus signal is fed into the OR switch. The OR switch invents the plus signal to minus (wedge) thus making the latch back minus which turns off the latch.



Here is still another type of latch. It has a normal latch back but two different means to set it. It can be turned on by satisfying the upper AND switch (heavy line). It can also be turned on by either the - Reset ROAR E signal or by the - Start Reset signal. Either one of these signals satisfies the rightmost OR switch. The dash above the latch name means NOT, so the latch name is read: " NOT ROAR BIT E".



Two (of several) latches shown above are reset and set by the same signal. The reset – set signal is at the bottom. When the signal is present, one of the legs of all latch back AND switches becomes negative because the signal is a minus signal. At the same time this signal feeds into an inverter. The inverter delays the signal for approximately 30 ns. When the plus signal leaves the inverter 30 ns later, the original minus signal has ended and the latches are set (provided that the other conditions + MOD. Bit 8 and + MOD. Bit P are present).



This latch is set by satisfying the upper most AND switch (heavy line). Its latch back, however, goes to two AND switches. This means that it can be turned off only by disabling both latch back AND switches. If, for example, the signal "+ Sec" goes minus, the first AND switch is not satisfied any more but the latch is not turned off because the other latch back AND switch keeps it turned on. So, both must be disabled.

5.4.2 How to Interpret a CAS Sheet

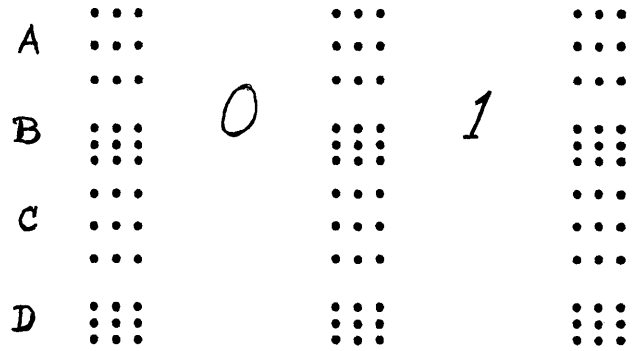
- CAS refers to "Controlled Automated System". The CAS sheets are produced by a computer according to a specific program.
- The CAS sheet shows exactly how a certain operation is accomplished by the micro-program.
- Normally, one complete operation is shown on each CAS sheet. All of the necessary micro-instructions of the operation and the sequence in which they follow each other is shown on one sheet.
- When the micro-program uses a Subroutine, the Subroutine is shown on an extra sheet.
- All lines, that leave a CAS sheet to enter a Sub-routine are labeled with the Sub-routine sheet number and a location code.

Description

CAS Sheet Title Block

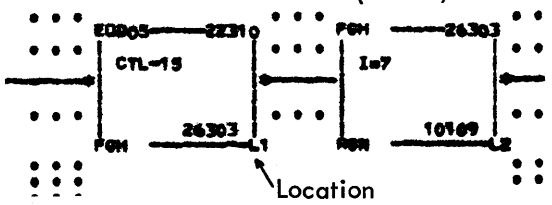
The all important CAS sheet number (Figure 5-10 is found at the right side. The title (subroutine STR+1 4K No. 1) shows what is actually accomplished by the instructions on the CAS sheet. Use the part number to order a new CAS sheet. The EC level indicates which version of the sub-routine is shown on the sheet. The date of change (or changes) shows when the sheet was last updated.

Page Coordinates. The CAS sheet is subdivided by a system of coordinates. Vertical columns are numbered from zero to nine (From left to right) and horizontal rows are labelled A to S (From top to bottom). (The characters O and I are omitted to avoid confusion.)

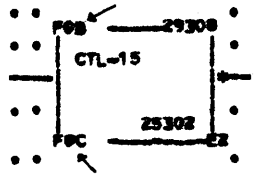


Micro-instruction Representation. The micro-instructions are represented as square frames. At the lower right corner of each frame is it's location code. For example:

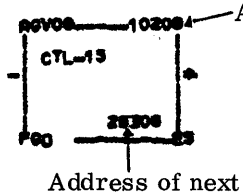
The micro-instruction CTL-15 (Control 15) is located in Row L column 1 (Arrow).



In both upper and lower left corners of each frame are symbolic addresses. These addresses are obsolete. Ignore them.



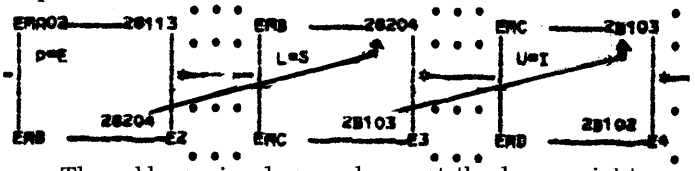
In the upper right hand is the address of the current instruction. The address 10208 reads out the CTL-15 instruction from the ROS. It is the instructions own address.



Address of current Instruction

Address of next sequential instruction

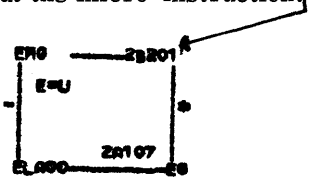
The address of the next Sequential Instruction is a part of each individual micro-instruction.



The address is always shown at the lower right hand side of each frame. As long as no branch occurs, each micro instruction causes the next following instruction to be read out. (Arrows) The former NSI address becomes the current address of the next instruction.

Address Bit Configuration in ROAR

An analysis of an address such as 2B201 gives the exact bit configuration which must be in the ROAR in order to read out the micro-instruction.



The first digit designates the Module in which this micro-instruction is found (Figure 5-11). The second digit (Hexadecimal representation) designates the Word (Inner or Outer) of the tape in the Module. The field selects one of the three instructions in the Word. The last two digits represent the decimal value of the last 4 bits (the position) in the ROAR. Block and position together determine the tape number.

Micro-program Branching

When the position (last 4 bits) of an address is changed, a different tape is selected. The micro program often changes these last 4 bits (upon a certain condition) in order to branch to an instruction other than the normal next sequential.

Changing the last 4 bits is affected by the USE-micro instructions. The 3 different USE-instructions, Use AND, Use OR, and Use Execlusive OR are always represented as UA, UO, and UX respectively.

A Use AND T instruction is performed in frame 1 (Figure 5-12). The content of data register T is ANDed with the last 4 bits of the NSI (IN the instruction). Following is an example which shows how the ANDing is accomplished:

8	4	2	1	Bit Value
1	1	1	1	Bits in DR-T
1	0	0	0	Bits (result)
8	4	2	1	Bit Value
1	0	0	0	Bits in Mask

Mask of NSI = 8
(Last four bits)

It is assumed that the content of DR-T is 15. Notice that the result is 8, that is, the same as before. Therefore the next instruction is the UOT (arrow).

The result of the ANDing is always 8 as long as there is an 8-bit in DR-T. This fact is indicated by the additional mask picture X NNN in frame 1. Let's assume that the content of DR-T is 7. The result is zero. Therefore the next instruction is I = 4 (Frame 6).

8	4	2	1	Bit Value
0	1	1	1	Bits in DR-T
0	0	0	0	Bits (result)
8	4	2	1	Bit Value
1	0	0	0	Bits in Mask

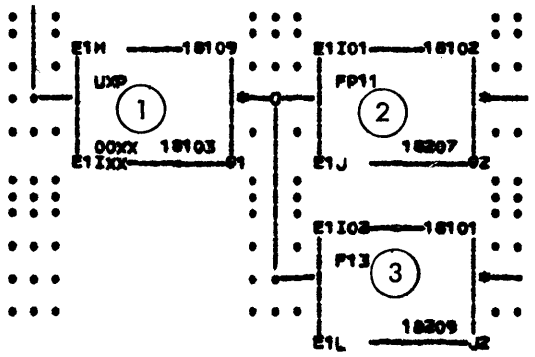
Mask of NSI = 8

In frame 2 the content of DR-T is ORed with the mask of the NSI (in the instruction). It is again assumed that DR-T contains 15. The result is 15. Therefore, the instruction in frame 7 is performed next. Notice that the result is always 11 as long as DR-T contains no 4-bit. If that is the case, the instruction in frame 3 is performed next.

8	4	2	1	Bit Value
1	1	1	1	Bits in DR-T
1	1	1	1	Bits (result)
8	4	2	1	Bit Value
1	0	1	1	Bits in Mask

Mask of NSI = 11

Here is an example of an Exclusive OR operation.



The content of DR-P is exclusively ORed with the mask of the NSI. It is assumed that DR-P contains 1. The result of the Exclusive ORing is 2. Therefore the instruction in frame 2 is performed next.

8	4	2	1	Bit Value
0	0	0	1	Bits in DR-P
0	0	1	0	Bits (result)
8	4	2	1	Bit Value
0	0	1	1	Bits in Mask

Mask of NSI = 3

	CHANGE	DATE	CHANGE	Part No	Logic No C3
				Title 2020 BUS, MODIFIER	

Figure 5-9 SLD Heading Block

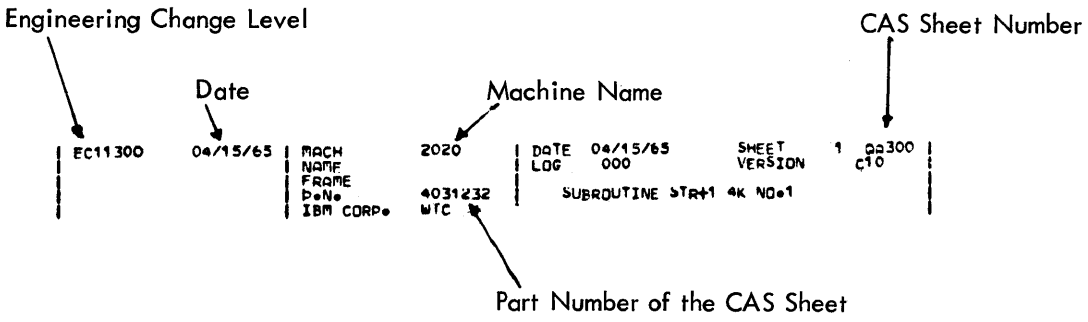


Figure 5-10 CAS Sheet Title Block

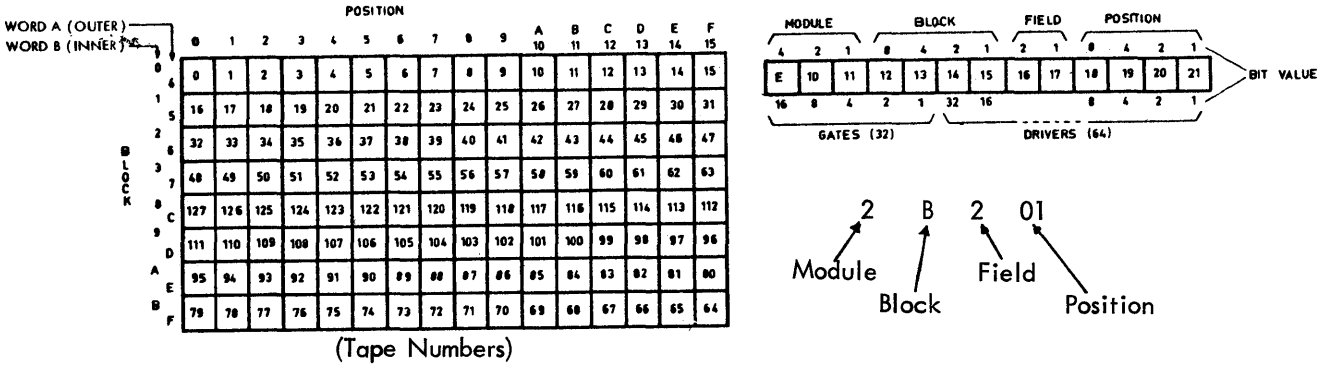


Figure 5-11 Tape Number Identification

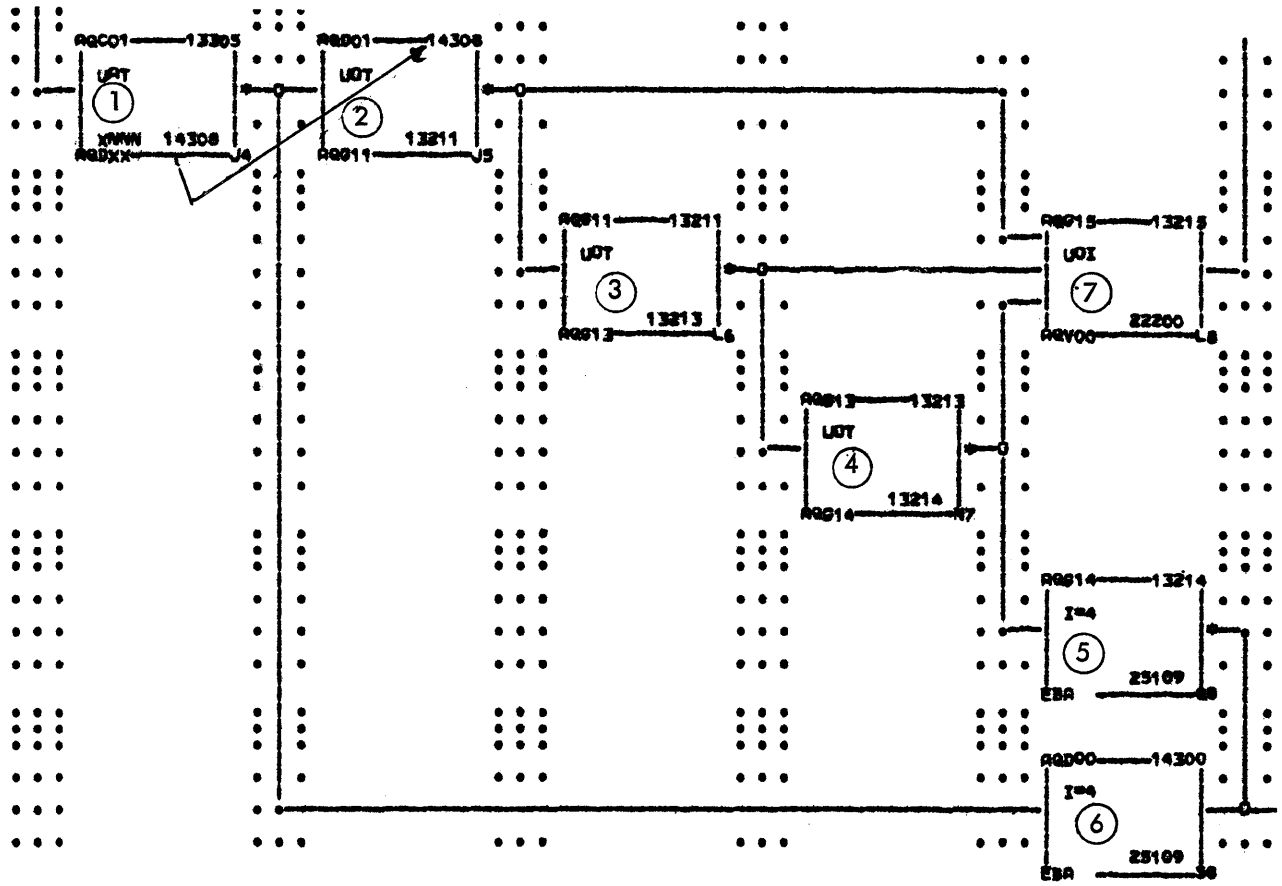


Figure 5-12 Branching Instructions

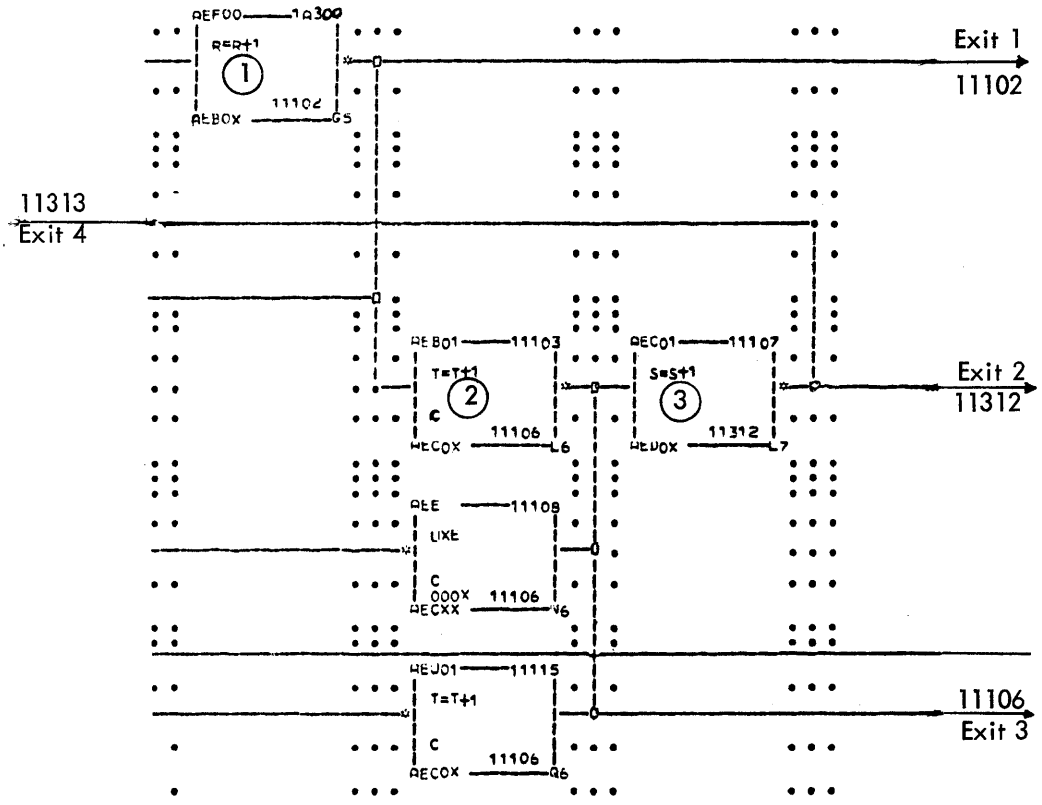


Figure 5-13 Micro-instruction Branching

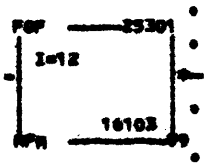
The mask picture in frame 1 (00XX) indicates that DR-P contains either a 1 bit or a 2 bit only. If it contains a 2 bit, the result is 1 and the instruction in frame 3 is performed next.

A branch in the micro-program can also be effected by a carry. In Figure 5-13, frame 1, the content of DR R is increased by 1. As long as the content of DR-R is not increased beyond 15, no carry occurs and the program continues with the next instruction at location 11102 (Exit 1).

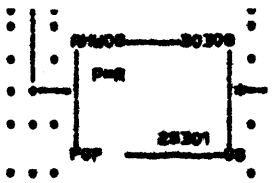
When DR-R contains 15 and is further increased by 1, a carry occurs. The carry increases the value of the last 4 bits (the mask) of the NSI by 1. Thus the next sequential address is 11103 instead of the normal 11102 and the instruction in frame 2 is performed. As this is a T + 1 operation, a carry may again occur and the NSI is increased from 11106 to 11107. Or, if no carry occurs the program continues with Exit 3. The same occurs with the S+1 instruction in frame 3.

Instruction Designation

The instruction I = 12 is a representation for "Move 12 into DR-I".

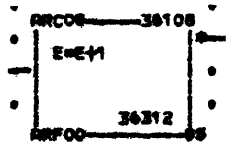


The instruction I = 12 should actually be shown in this manner: I ← 12. However, the printer which prints the CAS sheets has no arrow. Therefore, an equal sign (=) is used instead of an arrow (when I equals 12, it must be the same as 12, that is, it must contain 12).



P = R means: move the content of DR-R into DR-P (P ← R).

When reading instructions, such as these, always imagine the equal sign as being a left pointed arrow.



The instruction in the above frame says: "Increase the content of DR E by one". If the content of DR-E was 4 for example, it would be 5 at the end of the E = E+1 operation.

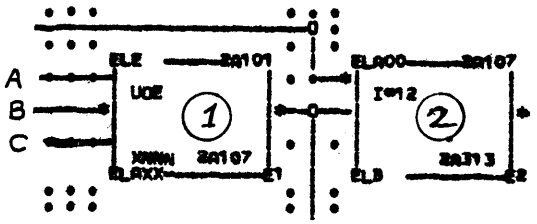
CAS Sheet Interconnections

When a line leaves a CAS sheet, it is labeled with the number of the CAS sheet to which it goes.

QA301 . E2A

CAS sheet No. Loc Code

In this example, the CAS sheet number is QA301 and E2A is the location code. The location code shows that the line enters the instruction which is located in Row E column 2 and Entrance A. When several lines lead to a micro instruction the entrances are labeled A, B and C.



In frame 2 the A and the B entrance is used. Sometimes a leaving signal is labeled with a second number. This number is the Return Number.

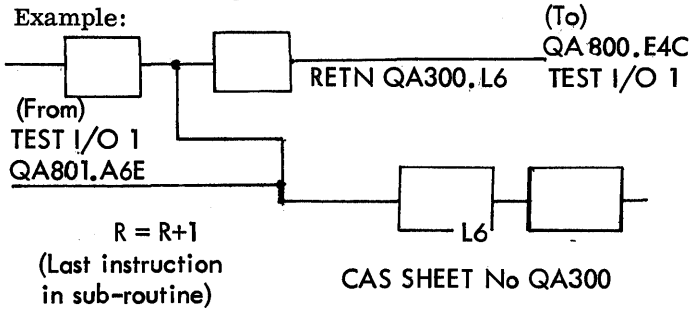
QA060 . C25 ← 1. Number

Identity Table U1

RETN QN204 . N1B ← 2. Number

Often a line leaves a sheet to go to a Sub-routine. If this line leaves the Sub-routine to return to the page from which it came, the Return Address is shown on the leaving signal line as a second number. When interpreting a CAS sheet, it is not always necessary to see the Sub-routine itself because the name of the Sub-routine indicates what is being done with the signal. Thus, checking of the Sub-routine sheet, only for address finding purposes, can be avoided.

Example:

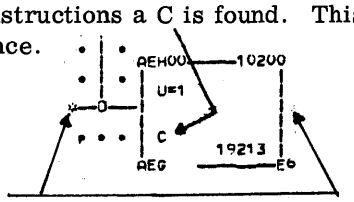


The leaving line goes to the Sub-routine "Test I/O No. 1". The Return Address indicates that the signal comes back to this sheet (QA300) and enters the instruction in frame L6.

As an additional aid, the last operation within the Sub-routine is shown near the return line. This helps in back-tracing the signal to the point where it leaves the Sub-routine since the routine may have a large number of exits.

Miscellaneous Designations

In some instructions a C is found. This is of little importance.



The C indicates a fixed address that may not be changed by an engineering change. Often instructions may be shifted to other addresses, however.

The asterisçs, found everywhere in the CAS sheets, are also unimportant. They mark the beginning of a network. (Concerns the designer, only)