amdahl™ 470/6

machine reference manual

G009-1

The Amdahl logotype (amdahl) and "Amdahl 470" are trademarks of Amdahl Corporation.

AMDAHL 470/6 SYSTEM
MACHINE REFERENCE MANUAL

PREFACE

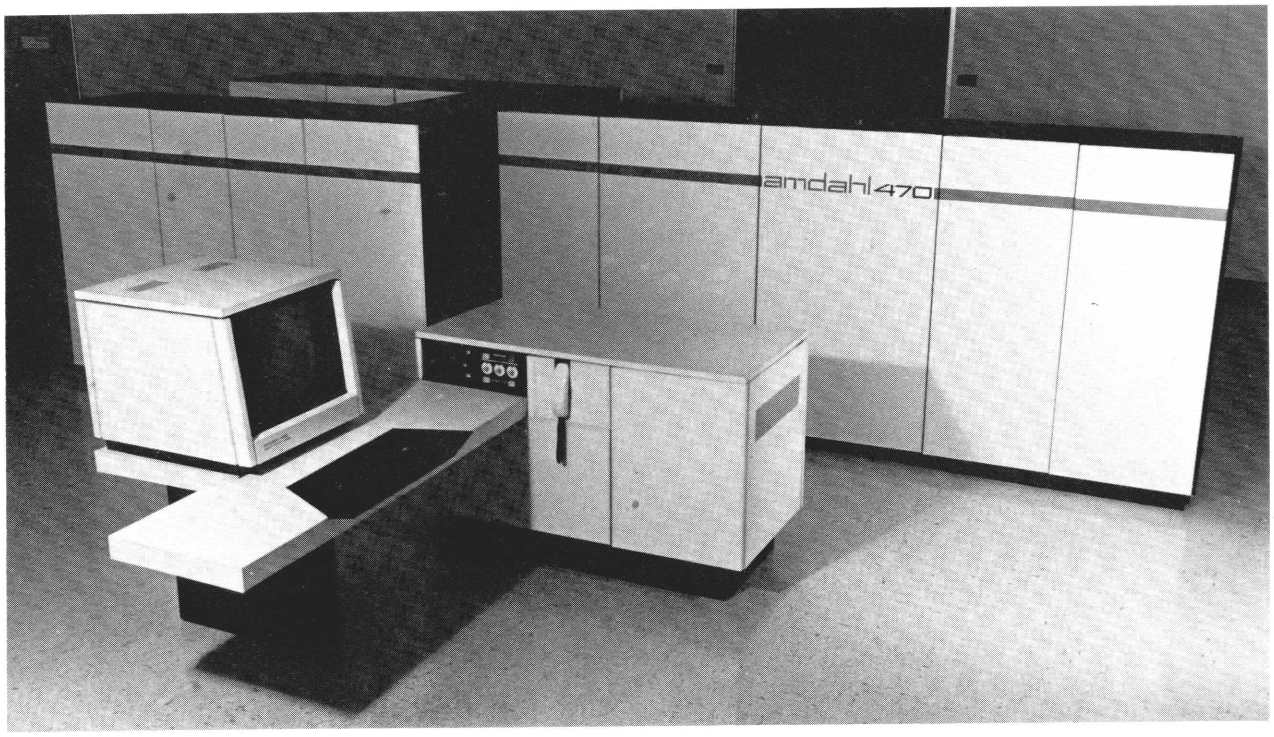
The *AMDAHL 470/6 System Machine Reference Manual* is intended as a reference document for programmers requiring a knowledge of the operating and functional characteristics of the 470/6 System.

The AMDAHL 470/6 Computing System is designed to execute the total IBM System 370 instruction set for real-memory processing, thereby providing complete compatability with this system. This document describes the AMDAHL 470/6 System and features additional to those available in S/360. Functions common to both systems are detailed in the publication *IBMSystem/370 Principles of Operation* (GA22-7000).

© AMDAHL Corporation, Sunnyvale, California 1973

CONTENTS

ILLUSTRATIONS



AMDAHL 470/6 System

1. AMDAHL 470/6 FUNCTIONAL OVERVIEW

The AMDAHL 470/6 Computing System provides very powerful, general-purpose capabilities for performing sophisticated data processing tasks. The innovative design of this system—the combination of subnanosecond logic with large-scale integrated circuit (LSI) technology, establishes the AMDAHL 470/6 as the first fourth-generation computing system of its magnitude.

SYSTEM COMPONENTS

The major components of the AMDAHL 470/6 System are the central processing unit (CPU), main storage, input/output channels, and the system console.

Eight possible main-storage configurations provide 1,048,576 to 8,388,608 bytes of directly-addressable data storage. Main-Storage cycle time is equal to 16 CPU cycles. The effective cycle time of the main-storage unit is statistically reduced by the overlapping (interleaving) of its operations and by channeling data through a high-speed buffer containing up to 16,384 bytes. The buffer can transfer eight bytes of data to the CPU in 2 CPU cycles.

The AMDAHL 470/6 central processing unit provides arithmetic and logic capabilities as well as controls for storage and channel functions. The operation of the various units controlling these functions is overlapped, allowing as many as five instructions to undergo some phase of execution concurrently. Data flow between the CPU components is shown in Figure 1.1.

As many as 16 input/output channels (1,024 subchannels) can be included as part of the AMDAHL 470/6 System configuration. Each channel incorporates System 360/370 input/output interface features, allowing the attachment of devices compatible with this interface. There is no design restriction on the configuration of selector, block-multiplexer, or byte-multiplexer channels, although operating system requirements limit this flexibility.

The system console includes an independent console processor, operator control panel, CRT display unit, and input/output control keyboard for performing system operation and maintenance functions. The system console provides a dual interface with the CPU, allowing three distinct modes of console operation: OS/370 device support, system control mode, or maintenance mode.

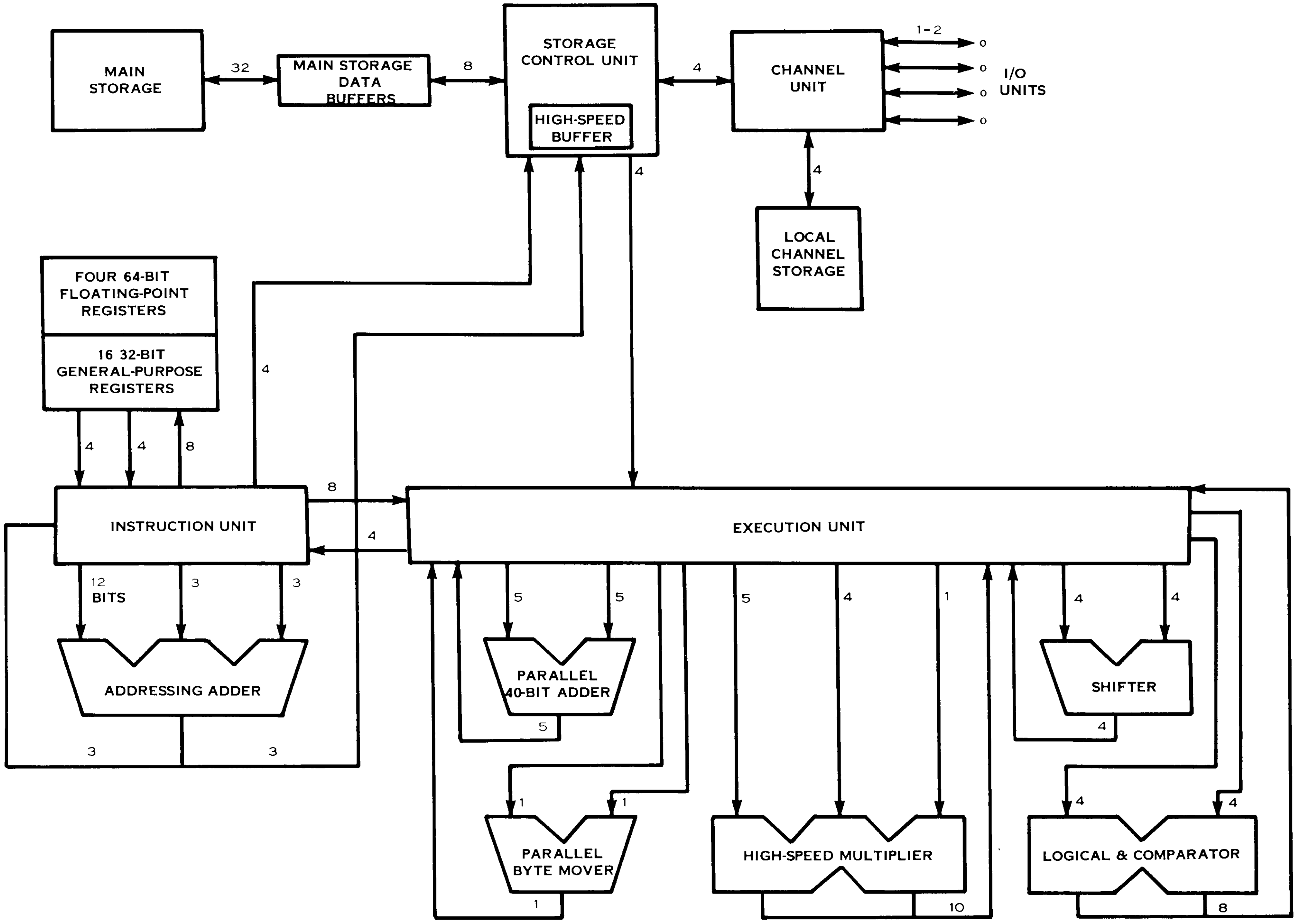


Figure 1.1 AMDAHL 470/6 Data Flow

PROGRAMMING CONSIDERATIONS

The AMDAHL 470/6 System implements the S/370 universal instruction set, including all instructions needed for real-memory processing.

System software support is provided through a slightly modified version of OS/360. Two control programs are available with this operating system: MFT (multiprogramming with a fixed number of tasks) and MVT (multiprogramming with a variable number of tasks). See the *AMDAHL 470 Operating System Support* (G201) document for details.

RELIABILITY AND MAINTAINABILITY

The AMDAHL 470/6 System offers advanced reliability and maintainability through its sophisticated design, LSI technology and error handling procedures. Highlights of these features include:

- Through LSI technology, a higher level of integration (density) than third generation technology, requiring fewer packaged components and connectors;
- Extensive hardware checking;
- Error checking and correction (ECC) for main storage, permitting correction of all single-bit failures and detection of all double-bit and most multiple-bit failures;
- CPU retry, allowing automatic hardware retry for most failing instructions without programming assistance;
- Hardware command retry for all channels, in addition to the software retry supported by OS/360;
- An independent console processor, enabling a failing main computer to be diagnosed by a working computer;
- Recovery Management Support (RMS), a software feature that minimizes the effects of failure through program damage assessment and selective job termination;
- Extensive machine-check facilities and processor logout, aiding in the rapid isolation and repair of failures;
- The ability to partition the high-speed buffer, bypassing a failing area and increasing system availability.

The AMDAHL hardware and operating systems are supported by three levels of AMDAHL personnel: customer-site software and hardware support representatives, the plant-site field support center, and the combined resources of manufacturing, engineering, and programming personnel. The AMDAHL field engineer has the ability to control and analyze customer problems remotely using a CRT terminal connected to the customer's system.

SYSTEM COMPATIBILITY

The AMDAHL 470/6 System is functionally identical to S/360 and S/370 real-memory systems. Users of these systems can advance to the AMDAHL computers with minimal transition considerations. The AMDAHL 470/6 System can execute OS/360 with only those modifications normally required between S/370 models.

Removal of ASCII Mode

S/360 allows its users to decide whether they want ASCII code or EBCDIC code generated for decimal results. If PSW bit 12 is set to one, ASCII mode is indicated.

The AMDAHL 470/6 System (like S/370) does not allow this option. In S/370, PSW bit 12 controls the system mode of operation and a program interruption for specification exception occurs if a S/360 program has the ASCII-mode bit set. All instructions that depend on the value of PSW bit 12 when they are executed in S/360 are executed in EBCDIC mode.

Invalid Decimal Signs

The removal of the ASCII-mode option also affects the handling of invalid decimal number signs. In the AMDAHL 470/6 System, if an invalid sign is noted during a decimal arithmetic operation, the operation is suppressed (instead of terminated, as in S/360). Specifically, the following action is taken:

Sign Bits are:	*Digit Bits are:*	*Instruction is:*
Valid	Valid	Completed
Valid	Invalid	Suppressed
Invalid	Valid	Suppressed
Invalid	Invalid	Suppressed

SUMMARY

The AMDAHL 470/6 System features and component relationships are summarized in Figure 1.2.

STORAGE FEATURES	
Main Storage (1)	High-Speed Buffer
Capacity: 1 - 8 Megabytes Cycle time: 16 CPU cycles Bytes per access: 32 Access time: 12 CPU cycles	Capacity: 8,192 or 16,384 bytes Cycle time: 1 CPU cycle Bytes per access: 4 or 8 Access time: 2 CPU cycles

CENTRAL PROCESSING UNIT FEATURES	
S/370 Universal Instruction Set	Extended-Precision Floating-Point
Instruction Overlap	High-Speed Multiply
Byte-Oriented Operand	Control Registers
Store/Fetch Protection	Direct Control (2)
Interval Timer	System Control with Display Console
Time-of-Day Clock	Main Storage Error Check & Correction
Storage Reconfiguration Control	Instruction Retry

CHANNEL FEATURES

Channel	Maximum No. Allowed (3)	Max. Transfer Rate (4)	
		1-byte Bus	2-Byte Bus
Byte Multiplexer	16	—	—
Burst Mode	—	2Mb/sec	4 Mb/sec
Multiplex Mode	—	135 Kb/sec	500 Kb/sec (6)
Selector	16	2 Mb/sec	4 Mb/sec
Block Multiplexer	16	2 Mb/sec	4 Mb/sec

Data-In/Data-Out (5)	Command Retry
Disconnect-In	

SYSTEM CONSOLE	
Independent Processor	16K Processor Storage
Dual Interface with CPU	Disk Storage
Operator Control Panel	Communications Modem
CRT Display Unit	

NOTES:

1. Odd-number models are 2-way interleaved: even-number models are 4-way interleaved.
2. Read/write direct and external interruptions.
3. As many as l,024 subchannels can be assigned, up to 256/multiplexer-channel; up to 16 total channels in any configuration.
4. Kb=l000 bytes: Mb=1 million bytes. Transfer rate is affected by cable length.
5. All data rates (except 1-byte-bus byte multiplexer) assume use of this feature.
6. Byte multiplexer 2-byte-bus rate is for block of 4 bytes.

Figure 1.2 Summary of AMDAHL 470/6 System Features

2. MAIN STORAGE

The storage facilities of the AMDAHL 470/6 System consist of three functionally separate entities: the main-storage unit, the storage-control unit (S-Unit), and the high-speed buffer contained in the storage-control unit (see Figure I.I). The S-Unit handles most logical operations for the main-storage unit and high-speed buffer.

MAIN STORAGE

The modular structure of the AMDAHL 470/6 main-storage unit allows for eight different CPU/main-storage combinations.

Configuration	*Storage Capacity (Bytes)*
1M	1,048,576
2M	2,097,152
3M	3,145,728
4M	4,194,304
5M	5,242,880
6M	6,291,456
7M	7,340,032
8M	8,388,608

Data Transfer

As was indicated in Figure 1.2, the main-storage-unit cycle time is equal to 16 CPU cycles. The main-storage unit contains two logical 32-byte data buffers, one for input and one for output (Figure 2.I). When a data transfer command (such as fetch) is received, 32 bytes are moved into a data buffer (in this case the output buffer). This fetch operation requires 12 CPU cycles. An additional four cycles are then required to move these 32 bytes, 8 bytes at a time, into the high-speed-buffer. The data are then passed to the requesting unit.

The programming implications of this 32-byte block structure are discussed in Chapter 10.

Interleaving

The data transfer scheme just shown describes a nonoverlapped storage system. In practice, the data transfer rate (for multiple transfers from sequential main-storage addresses) is significantly improved by interleaving. Interleaving also allows several units of the CPU to access data concurrently.

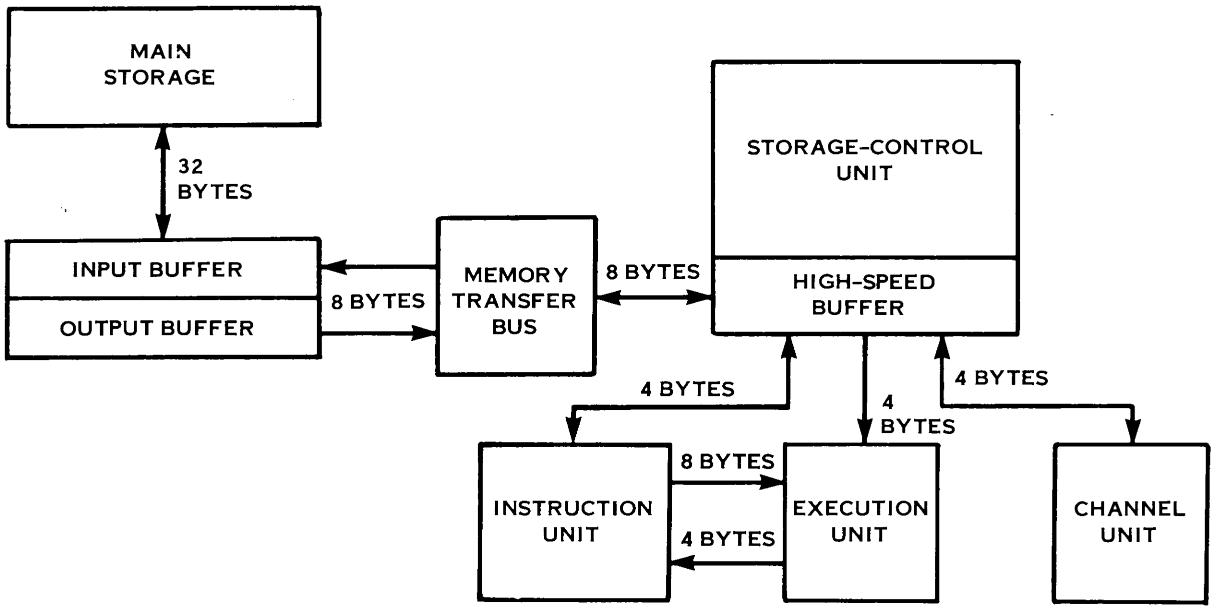


Figure 2.1 Data Transfer

Interleaving takes advantage of the modular structure of the main-storage unit to increase the number of independent storage accesses initiated in a single storage cycle. The even-numbered storage configurations (2M, 4M, 6M, and 8M) are four-way interleaved. The odd-numbered configurations (1M, 3M, 5M, and 7M) are two-way interleaved.

ERROR CHECKING AND CORRECTION (ECC)

The AMDAHL 470/6 System includes an error checking and correction feature that permits automatic correction of all single-bit failures in main storage and detection of all double-bit failures. The storage structure is such that it can operate correctly with any single data storage field-replaceable unit in error (provided no other errors exist). With this feature present the total number of expected system interruptions will be reduced.

When data are stored, an ECC code is substituted for the parity bits in the stored bytes. The group of 16 bytes (quadword) associated with a single ECC code is called an ECC block (Figure 2.2).

128 DATA BITS	9 ECC BITS

Figure 2.2 ECC Block Layout

If a single-bit error is detected when the data are fetched, the bit is corrected in the high-speed buffer. A buffer modification indicator is set and the erring main-storage location is updated with the corrected data the next time the buffer is stored. Program execution may continue without interruption, depending on interruption masking.

If the error occurs in the first quadword of the 32-byte data block, the correction in the high-speed buffer takes four cycles. Six cycles are required to correct an error in the second quadward and eight cycles if an error is present in both quadwords.

If a double-bit or multiple-bit error is detected, it is generally referred to the system for processing under program control. Correction under program control is done on a 16-byte data block basis.

BYTE-ORIENTED OPERAND FEATURE

Systems 360/370 place certain restrictions on fixed-length data fields located in main storage. These fields must begin on integral boundaries, meaning their storage addresses must be multiples of their length (in bytes).

Halfword (2 bytes)	Boundary must be a multiple of 2
Word (4 bytes)	Boundary must be a multiple of 4
Doubleword (8 bytes)	Boundary must be a multiple of 8

The byte-oriented operand feature of the AMDAHL 470/6 System removes this restriction. If the referenced data field is an operand in an "unprivileged" instruction, it can be located on any byte boundary with only a minor loss of performance (nominally 1/8 CPU cycle per operand storage reference).

Operands appearing in "privileged" instructions are not affected by this feature and must conform to boundary restrictions. Privileged instructions are those that can only be performed under system supervisor control (e.g., LOAD PSW, DIAGNOSE, STORE CHANNEL ID).

3. STORAGE-CONTROL UNIT

The operations of both the main-storage unit and the high-speed buffer are supervised by the storage-control unit (S-Unit). It also handles all references to storage from the CPU's instruction and execution units and from the input/output channel unit.

HIGH SPEED BUFFER

Physical Characteristics

The basic AMDAHL 470/6 System configuration includes an 8192-byte high-speed buffer (HSB) in the storage-control unit (Figures 1.1 and 2.1). The system also includes a buffer extension feature, allowing this area to be expanded to 16,384 bytes.

All processing of data into and out of the system goes through this high-speed buffer. If some data block is referenced frequently, it may be modified many times in the buffer before it is put back into the main storage unit.

Data are transferred between main storage and the high-speed buffer at a rate of eight bytes per CPU cycle. (The buffer cycle is the same as the CPU cycle).

Data are transferred between the HSB storage area and the various CPU units at a rate of four bytes per CPU cycle, or eight bytes per two cycles for doubleword accesses. For most applications using the 360/370 architecture, four bytes is the usual length of data and instructions.

Physically, the 8K high-speed buffer is divided into 128 32-byte blocks of primary buffer storage and 128 blocks of alternate storage. Each storage address maps into a single primary-block location and a single alternate-block location. After a given block is selected for input or output, this arrangement allows bytes to be selected individually.

This division of the HSB into primary and alternate storage areas also provides the buffer with a partitioning capability. This permits the bypassing of a portion of the buffer should a failure occur, thereby increasing system availability. Partitioning is two-way for an 8K buffer, four-way for 16K.

Buffer Tags

Each block in the high-speed buffer includes a tag containing identification and control information; thus there are 128 primary tags and 128 alternate tags. The tag bits and their usage are as follows (Figure 3.1):

- Block ID (12 bits). These bits, along with the HSB location address, define the block from main storage that currently occupies the HSB location associated with this tag. Corresponds to main-storage address bits 8-18 and 26.
- Address Parity. Three bits, identical to the parity bits contained in the three bytes (24 bits) required for this main-storage address. This field is used for address parity checking within the S-Unit.
- Key. Five bits and parity. The 4-bit storage-protect key associated with this main-storage address is stored into the HSB tag when the data are brought from main storage. The fifth bit of this field is used for fetch protect.

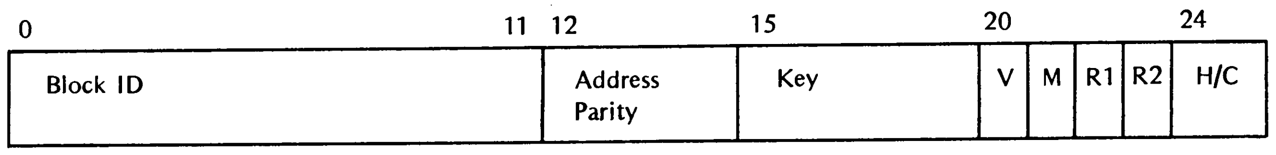


Figure 3.1 HSB Tag Format

- Validity. If this bit is one, the data in this block are a valid representation of their main-storage counterpart.
- Modification. If this bit is one, some of the data contained in this HSB block have been changed since being obtained from main storage. If this HSB location is needed to buffer data from a different MS location, it is not necessary to store the present data back into main storage if they have not been changed.
- Requester (R1, R2). When the R1 bit is zero, the data in this block were obtained from main storage as the result of a channel unit request. If R1 = 1, the I-Unit or E-Unit was the requester.

R2 is only significant if the data in this block are CPU data (R1=1). If so, then R2=1 indicates the CPU was in the supervisory state when the request was issued. Otherwise the CPU was in the problem program state.

- Currency of use. The "hot/cold" bit is separate from the tag buffer and the data HSB. If set to one, it indicates that the primary block for this HSB location has been accessed since the alternate (primary hot). Whenever a block must be purged to make space for different main-storage data, the H/C, modification and requester bits are used to determine whether the primary or alternate block should be purged.

Figure 3.2 shows how a 24-bit storage address, generated by the CPU or channel unit, is mapped into a buffer storage location.

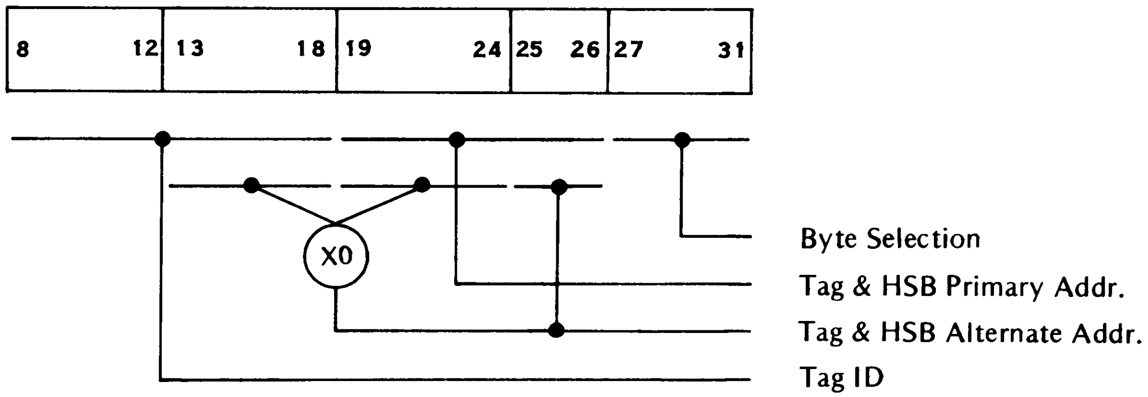


Figure 3.2 Mapping of 24-Bit Address into 16K Buffer

Bits 8-26 define a main-storage address. Bits 27-31 point to a specific byte within the 32-byte block. Bits 8-26 are defined as follows:

Bits	*Meaning*
8-18	Tag ID. Determines 8K block of main storage (out of possible 128 blocks per megabyte).
13-18	Exclusive OR of these bits, concatenated with bits 26-26, determines address of alternate buffer storage location.
19-26	Primary buffer storage address.

NOTE: For an 8K buffer, bit 26 is part of the Tag ID and not part of the buffer address. Main storage interleaving is controlled by bits 25-26.

STORAGE-CONTROL UNIT OPERATION

Addressing Paths

The primary route for addressing the HSB and main storage consists of the instruction unit's effective address register (EAR), the S-Unit's compare register, the HSB tag ID and the main-storage address register (MSAR). There are also four ports (Instruction-Fetch, Operand, Channel and Prefetch) used as holding registers for addresses of data that must be accessed from main storage (Figure 3.3).

When a buffer request is honored, the requested address is gated through the EAR into the compare ıegister and the appropriate port. The HSB tag is accessed to determine whether the requested data are currently in the HSB. If they are in the HSB, are valid, and the storage-protect key matches, the data are gated to the requester. If the requested data are not in the HSB, main storage must be accessed.

Before addressing main storage for the requested data, the modification bit in the tag of the HSB block to be replaced (primary or alternate) is tested. If the data have been modified, they must be backstored before replacing them with the requested data. The main-storage address to which these data must be backstored is gated from the Tag ID (bits 0-11) and the compare register to the MSAR, and four CPU main-storage-write cycles (four CPU cycles) are initiated. During these operations, the requested main-storage address is retained in the port associated with the requester and the requested block is read into the main-storage output buffer.

When backstoring is completed or not required, the contents of the compare register are gated to the MSAR, the main-storage read request is initiated, the HSB block to be replaced is invalidated, and four CPU main-storage-access cycles (four CPU cycles) are initiated to load the new HSB block. After the last access is completed, the originally requested address is gated from the port back to the EAR and compare register; the HSB is accessed and the data are gated to the requester.

Control Logic

The storage-unit functions are controlled as a 4-step pipeline. Each step is independent of the others, allowing as many as four buffer access cycles to be in progress at any one time. The steps (CPU cycles) and their primary functions are as follows (see also Figure 3.3):

A/P (Address and Priority Cycle)

- Determine request priority. The order of request priorities is: error correction (ECC), S-Unit-generated requests, channel (high priority); CPU (high priority); CPU (low priority); channel (low priority); prefetch.

- Decode and gate HSB addresses.

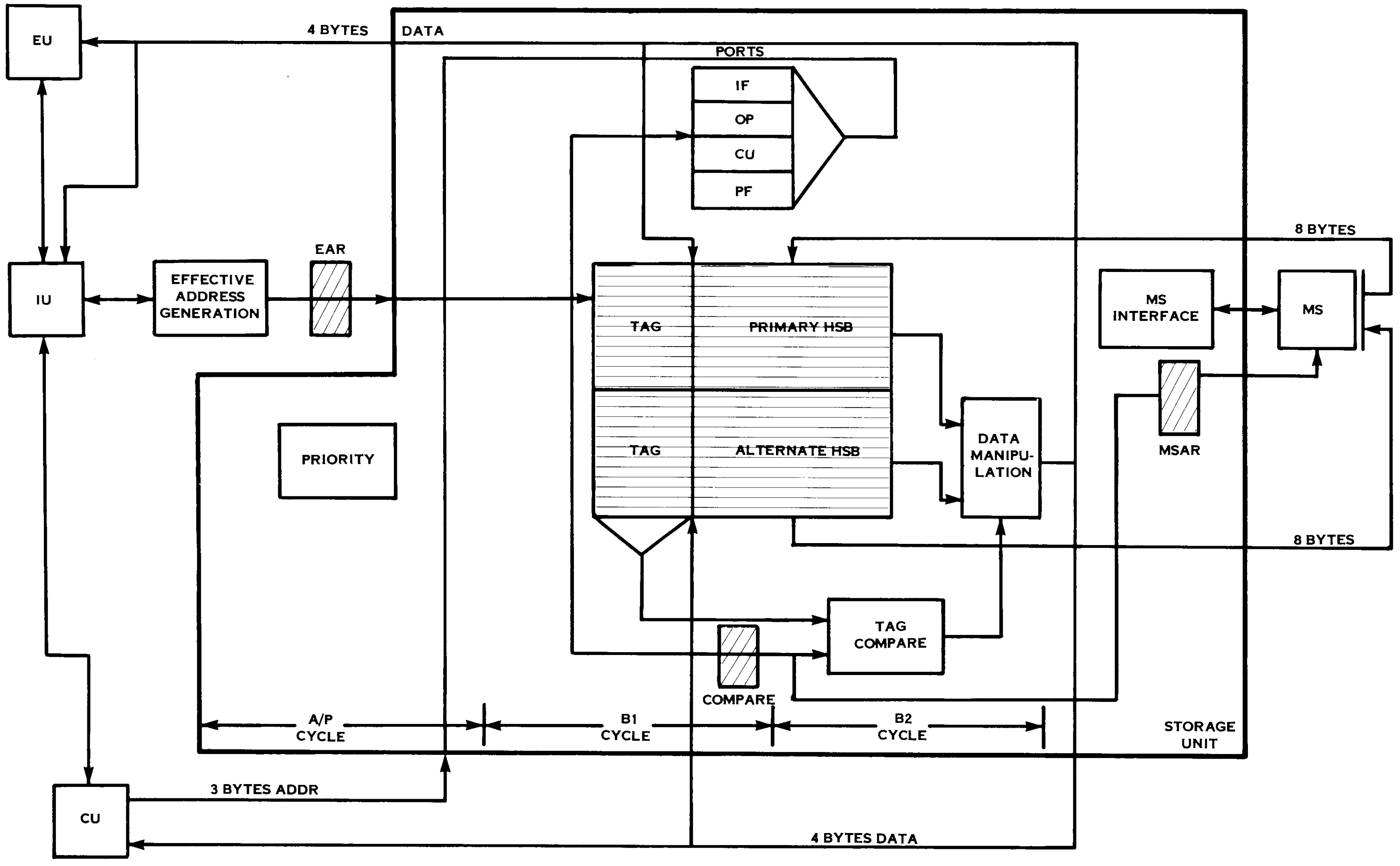


Figure 3.3 S-Unit Operational Flow

B1 (Buffer Cycle 1)

- Load EAR with selected address.
- Address primary and alternate HSB tag and data.

NOTE:If the data block to be accessed crosses a storage block boundary, a second S-Unit request is generated.

B2 (Buffer Cycle 2)

- Set port and compare registers;
- Compare tag to determine block present or missing, storage-protect-key check, and block valid;
- Conditionally gate HSB data or set up main-storage access.

R (Results)

- Set output register or generate main-storage-access request.

The possible overlapping of functions is indicated by Figure 3.4.

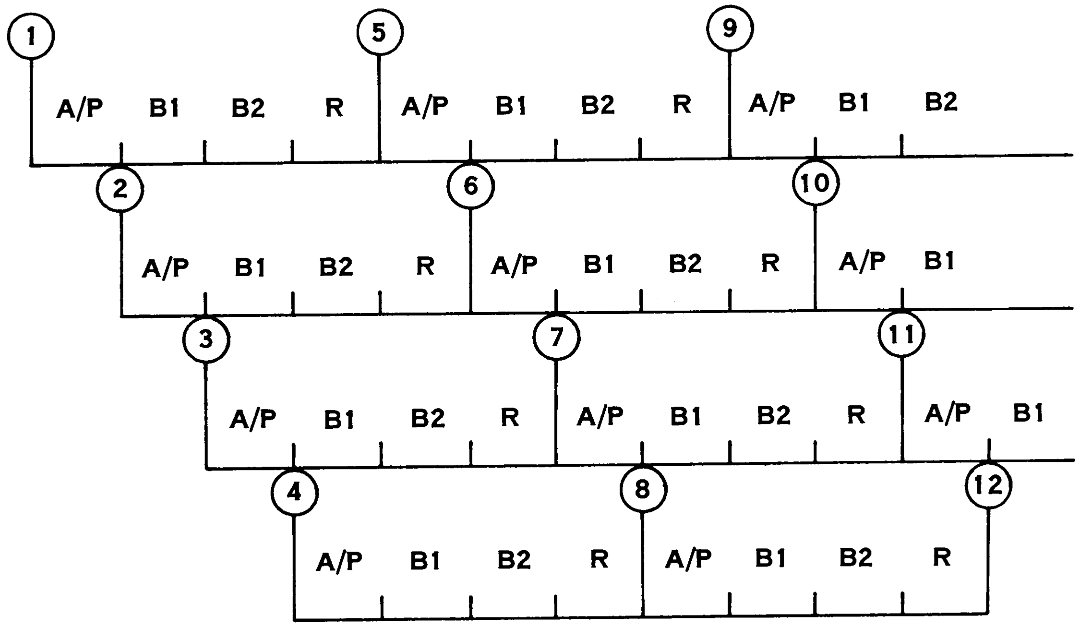


Figure 3.4 HSB Cycle Overlap

Main-Storage Access

When a request is made for a data location not in the HSB, the S-Unit must access main storage for the data. To do this, Move-In or Move-Out requests are entered into the pipeline. During the time necessary to actually move the data to or from main storage, the originally requested address is held in the port associated with the request, and no further requests are honored from that requester. Requests from other units and S-Unit-generated requests, however, continue to be processed by the pipeline. When the data become available from main storage, the main-storage unit signals the S-Unit with an identifier indicating for which port the data were obtained. The S-Unit then stores the data into the HSB and sets the appropriate tag bits. The specific operations performed are as follows:

- Storage addresses in the buffer are selected on a 32-byte block basis.
- Eight bytes of data are fetched from main storage through the Memory Transfer Bus (MTB) and stored in the high-speed buffer's data register.
- Nine ECC bits are sent to the S-Unit for each 16 bytes (see discussion of ECC in Chapter 2).
- The data are then moved to the buffer's store-select logic and written into the HSB. A copy is also sent to the ECC portion of the S-Unit so that single-bit errors can be checked for and corrected (if necessary) and parity can be generated.
- The eight parity bits resulting from the data are returned to the high-speed buffer for storage. The eight bytes stored in the buffer thus consist of the data transferred from main storage and the parity generated in the S-Unit.

A 32-byte block of data is moved into the high-speed buffer during four successive buffer cycles.

As these procedures indicate, the data section of the high-speed buffer communicates with both the S-Unit and main memory. The tag section of each 32-byte block communicates only with the S-Unit, however. There is no direct communication between the two sections of the high-speed buffer.

High-Speed Buffer Access

When data are transferred from the high-speed buffer to main storage, the following operations are performed:

- Data in the high-speed buffer are selected on a 32-byte block basis.
- Data are provided to the main-storage unit through the 8-byte wide MTB path.
- A copy of the data is provided to the S-Unit. The parity of the data is generated and checked against the parity that was in the high-speed buffer. In parallel with this action, the ECC check-bit codes are generated and sent to the main-storage unit.

A 32-byte block of data is moved to main storage in four consecutive buffer cycles.

4. INSTRUCTION UNIT

INSTRUCTION-UNIT PIPELINE

The main operations performed by the AMDAHL 470/6's instruction unit (I-Unit) are instruction fetching, decoding, and buffering; effective address generation; operand fetching; the issuance of instructions to the execution unit; result handling; and interruption handling. Figure 4.1 profiles these I-Unit functions. In this figure, each hash mark represents one CPU cycle and in general one function is performed per cycle. The functions in the sequence shown are actually "pipelined," meaning different functions can be processing different instructions concurrently. A given function can accept a new input every one or two cycles. In practice, the pipeline structure permits up to five instructions to undergo some phase of operation concurrently.

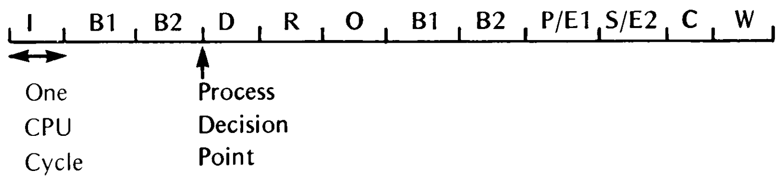
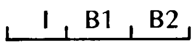


Figure 4.1 I-Unit Processing Pipeline Profile

Instruction Fetch



Instructions can be fetched from the storage-control unit's high-speed buffer at a maximum rate of one instruction word every two cycles. Data are fetched at a maximum rate of one word every cycle.

During the I, B1 and B2 cycles the address of the instruction word is formed in the I-Unit's effective address register (EAR), the word is fetched from the buffer and then stored in I-Unit local storage. Local storage consists of two 1-word buffers: the instruction word register (IWR) and the instruction buffer register (IBR). The contents of local storage are "bubbled up" on a "first-in, first out" basis to accommodate each instruction fetch into the IWR.

Checks performed during these three cycles include local storage, effective address register and high-speed buffer availability, and verification of the validity of accessed data.

An internal counter synchronizes the instruction fetching operation to the development of the instruction address in bit positions 40-63 of the program status word (PSW).

NOTE: If the desired instruction is not in the high-speed buffer when the I cycle begins, the B1-B2 operations will be delayed because the instruction has yet to be fetched from main storage. If the instruction is in the buffer, "prefetching" may be initiated by the S-Unit during the B2 cycle. This might consist of transferring the next block from main storage into the buffer, moving information out of the buffer, checking priorities, etc.

Instruction Decode Operation

| D | R |

In parallel with all other activity, an independent check is made for a pending operation (interruption, console request, machine check, etc.). If such an operation is pending, a hardware command is inserted in lieu of the instruction fetch at the "process decision point" shown in Figure 4.1.

During the data decode cycle, checks are made to determine whether the instruction is privileged, or whether a specification, addressing or protection exception should be recognized and encoded.

Then, on the R cycle, all index and base registers are accessed and their contents stored in the I-Unit's effective address operand registers. (This operation can be bypassed in favor of an execution unit result storage operation).

Operand Fetch

| O | B1 | B2 |

The 24-bit address of the instruction operand is calculated and placed in the EAR. The operand is requested from the high-speed buffer. Note in Figure 4.2 that even though the instruction address and operand address calculations both use the EAR, the pipeline structure is such that I-B1 and O-B1 operations do not coincide.

Up to two operand storage addresses are maintained for multiple accesses, making use of two I-Unit address registers. Multiple operand expansion also takes place in this phase; all instructions look equally simple to later execution phases.

A specification exception is recognized in the case of privileged instructions if the requested operand is not located at an integral boundary address. Addressing and protection exceptions governing the operand request are also recognized and encoded at this time.

In addition to fetching the operand(s) needed, this phase prepares the execution unit for the coming instruction. During the B2 cycle a priority check is performed to determine if the execution unit is available (i.e., if the previous instruction has been completed). This determines whether the next two cycles perform the P-S or E1-E2 functions.

Instruction	1	2	3	4	5	6	7
Cycles 1	I						
2	B1						
3	B2	I					
4	D	B1					
5	R	B2	I				
6	O	D	B1				
7	B1	R	B2	I			
8	B2	O	D	B1			
9	E1	B1	R	B2	I		
10	E2	B2	O	D	B1		
11	C	E1	B1	R	B2	I	
12	W	E2	B2	O	D	B1	
13		C	E1	B1	R	B2	I
14		W	E2	B2	O	D	B1
15			C	E1	B1	R	B2
16			W	E2	B2	O	D
17				C	E1	B1	R
18				W	E2	B2	O
19					C	E1	B1
20					W	E2	B2
21						C	E1
22						W	E2
23							C
24							W

Figure 4.2 Overlapped Instructions in Processing

First Execution Phase P/E1

If the P function is selected during the B2 cycle, a storage priority check is performed. Otherwise, the operands are transmitted into the execution unit in the first execution phase.

Second Execution Phase S/E2 C

The S function, if performed, writes result data to the buffer. Otherwise the E2 function transmits the second word of a doubleword operand (if necessary), and stages and checks the resultant data into the execution unit's result register.

Arithmetic exceptions are recognized and encoded. If a program exception is recognized at this time, control is transferred to the interruption handler.

Note that both the first and second execution phases are logically *pre*processing (except for stores). As far as the software knows, the instruction has not yet been executed.

Although instruction execution (E1-E2) is shown as a two-cycle operation, the actual time required depends on the specific instruction performed. See Chapter 11 for more precise guidelines.

Write Result

The results of the instruction execution are written to a general-purpose or floating-point register. The internal counter synchronizes the instruction completion to the contents of the PSW instruction-address field.

General Comments

The instruction pipeline is designed to favor RX-type instructions. As can be seen from Figure 4.2, the best throughput time is two cycles, assuming no interlocks within the pipeline (such as those caused by data or facility unavailability, multiphase operations, etc.)

All interruptions are precise. This means that the instruction-length code in bits 32-33 of the PSW will always be set for the proper instruction, when applicable, enabling the user to pinpoint more easily the instruction causing the interruption.

The programming implications of the I-Unit's pipeline structure are discussed in Chapter 10.

TIMING FEATURES

Instruction timing and elapsed time measurement are accomplished by the interval timer and time-of-day (TOD) clock.

Interval Timer

The interval timer has a resolution of 3.33 milliseconds and a clock cycle of 15.5 hours. It is a 32-bit binary counter residing at main-storage location 80 and is decremented every 3.33 milliseconds in bit position 23 (see Figure 4.3).

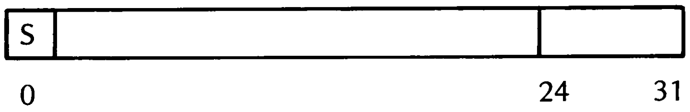


Figure 4.3 Interval Timer Bit Positions

An external interruption occurs when the interval timer's value changes from positive to negative. Both PSW bit 7 and bit 24 of control register 0 must be set to one for a timer interruption to be enabled. Bit 8 of the external interruption code is set to one when a timer interruption takes place.

Time-of-Day Clock

The time-of-day clock permits highly accurate lapsed time measurements (resolution of one microsecond) and time-of-day information. The clock cycle is about 143 years.

The clock is a 64-bit binary counter and is incremented by adding a one to bit position 51 every microsecond. The bit positions are numbered 0-63 as in an unsigned double-precision fixed-point number (Figure 4.4). The low order bits 52-63) are generally ignored.

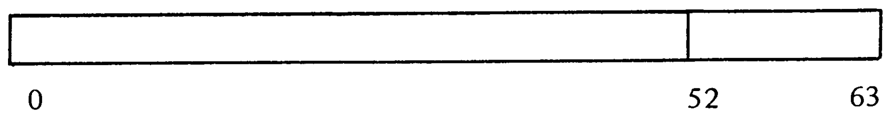


Figure 4.4 TOD Clock Bit Positions

No overflow is indicated as the clock value changes from a large positive number to a large negative number because of a carry from bit 1 into bit 0. No interruption is generated due to the overflow. A carry out of bit position 0 is ignored, and counting continues from zero.

The operation of the TOD clock is unaffected by normal activity in the system. The clock runs when the CPU is in the wait or stopped state or in the instruction-step, single-cycle, or test modes. A hard-stop condition, system reset or the IPL procedure also have no effect on the clock.

An operational TOD clock can be in one of three states: set, not-set, and error. When the power for the clock is turned on, the clock's value is set to zero, and it enters the not-set state. The clock enters the set state when SET CLOCK sets its contents. The clock can be placed in the set state from either the not-set or the error state. The clock enters the error state whenever it stops or misses a time increment, as when the power supply is disconnected temporarily or a malfunction is detected that might have affected the validity of the clock's value.

5. EXECUTION UNIT

EXECUTION-UNIT COMPONENTS

The execution unit (E-Unit) of the AMDAHL 470/6 System performs the instructions passed to it from the instruction unit. It contains 5 major functional components and 11 registers (Figure 5.1). Input data to the E-Unit pass through the logical and comparator (LUCK) component. Arithmetic operations are performed by the 40-bit carry-propagate adder, the 8-bit parallel byte mover, the 40-bit high-speed multiplier, and the 64-bit to 32-bit shifter. After manipulation, the result is stored in the result register, from which the information is returned to the I-Unit.

A new instruction can be transferred to the E-Unit from the I-Unit every two CPU cycles (depending on the execution time of the previous instruction). During execution the basic data transfer consists of gating a register through a functional unit (adder, multipler, or shifter) to a register in one cycle. The basic data path within the E-Unit passes four bytes in parallel with parity specification for each byte. Arithmetic operations also cause a residue "checksum" to be generated (used to check the 40-bit adder and the multiplier).

The storing and gating of information is managed by a control unit and the several registers within the E-Unit. The registers include the 8-bit I-register, four 32-bit working registers (1H, 1L, 2H, 2L), 8-bit B-register, 4-bit G-register, three 40-bit special registers (S, C, A) and the result register. There are also eight 4-byte "scratch" registers used by the I-Unit and E-Unit to execute certain instructions. Finally, there is a table lookup unit used in connection with the system's divide algorithm.

FUNCTIONAL COMPONENTS

Logical and Comparator Component

All input to the E-Unit passes through the logical and comparator component. This component is made up of a number of small independent functions (Figure 5.2). Each function works from common inputs, but may or may not give common outputs. The LUCK functions include:

- Logical functions: AND, OR, Exclusive OR.
- Through path provides a path to move two input operands unaltered to the outputs.
- Bit counter counts the number of leading zeros in the two input operands, using several counting modes.

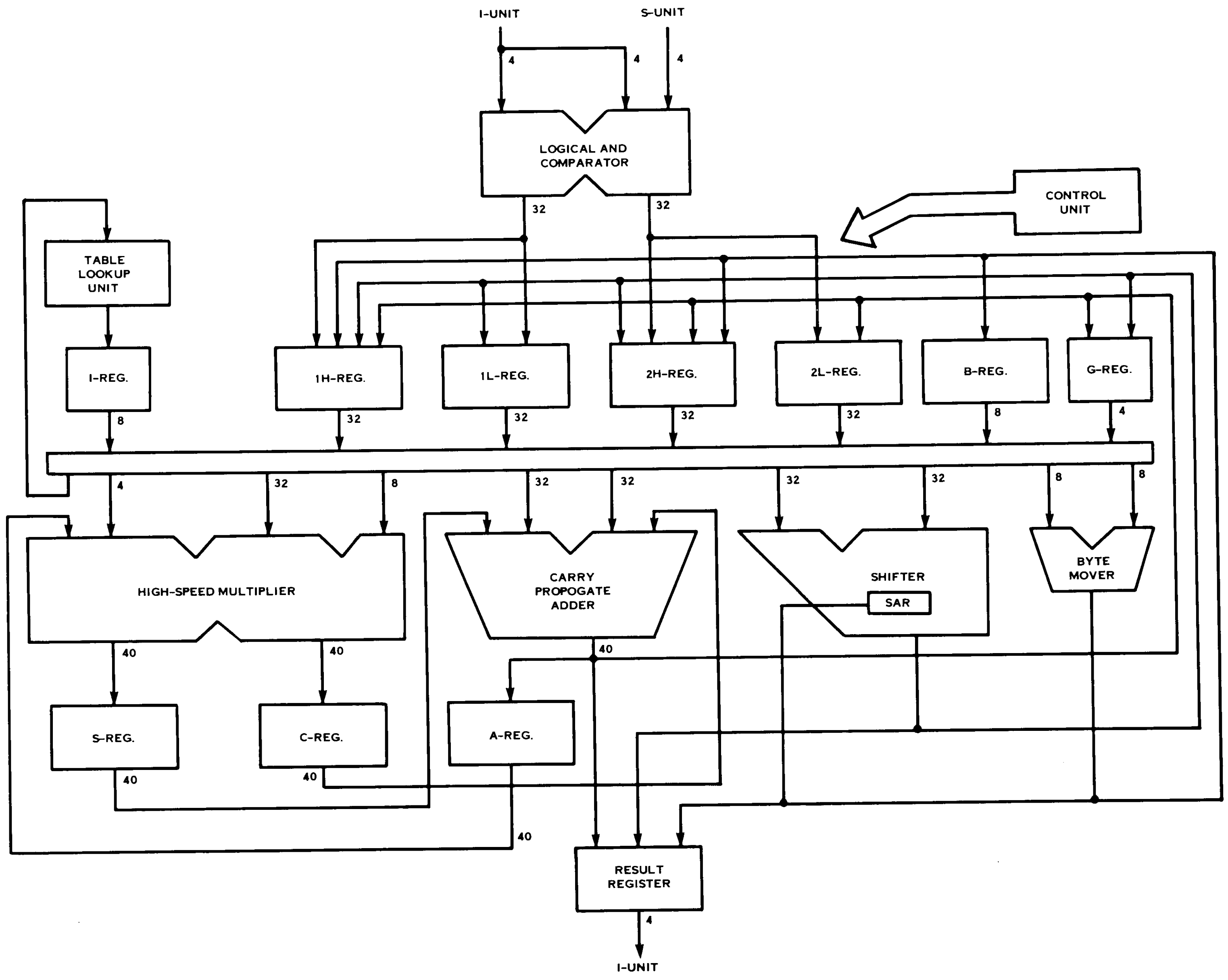


Figure 5.1 Execution-Unit Components

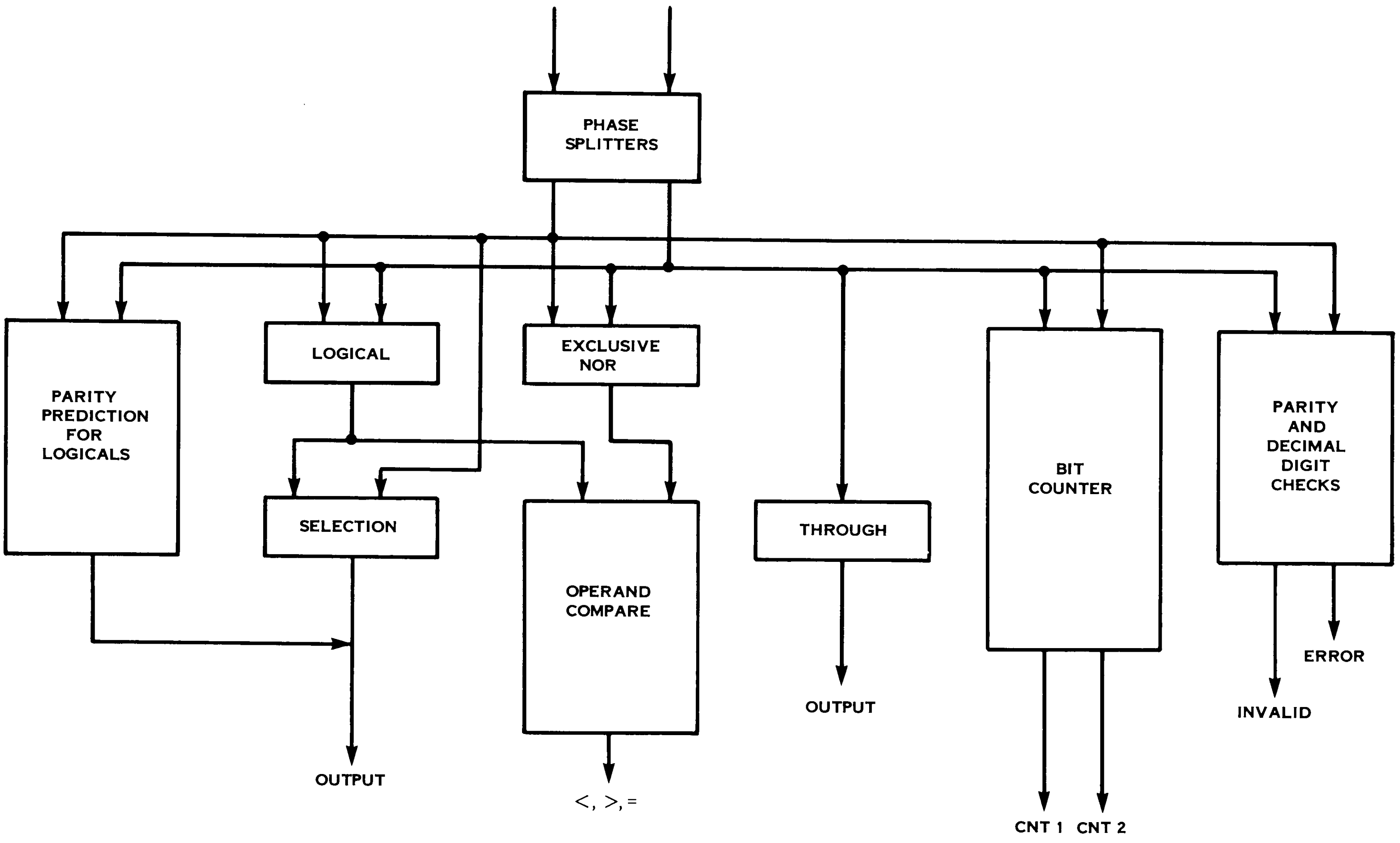


Figure 5.2 E-Unit Logical and Comparator Component

- Decimal-digit check checks the two inputs for valid decimal or sign digits. Whether the low-order digit should be a sign or decimal digit is controlled by a mode control.

- Operand comparison compares the magnitude of the first input operand with the magnitude of the second input operand. It provides a greater than, less than, equal to output. A test is also provided to predict arithmetic result overflow. This allows compares and arithmetic operations (except floating-point operations with unnormalized inputs) to set the condition code early.

- The inputs are checked for valid parity. Parity is predicted for logical operations. An independent method is used to predict the parity so that the prediction can be used to check the logical unit.

- The LUCK can also set the condition code for the TEST UNDER MASK instruction.

High-Speed Multiplier

The high-speed multiplier is used to execute fixed-point and floating-point multiply instructions. It can accept three inputs and produce two outputs that when added together will produce the following result:

$$(A \times B) + C = R_1, R_2 \qquad R_1 + R_2 = \text{Result}$$

where:

A = 32-bit number
B = 8-bit number
C = 40-bit number

To multiply two 32-bit numbers, the high-speed multiplier performs the following steps:

- The 32-bit multiplicand is multiplied by the 8 low-order bits of the multiplier and the 40-bit product stored in a result register.

- The 32-bit multiplicand is multiplied by the next 8 low-order bits of the multiplier, the product is shifted left 8 bit positions and added to the result register.

- The multiplicand is multiplied by the next 8 low-order bits, the product is shifted left 16 bit positions and added to the result register.

- The multiplicand is multiplied by the 8 high-order bits of the multiplier, the product is shifted left 24 bit positions and added to the result register to produce the final 64-bit result.

Carry-Propogate Adder

The carry-propogate adder is a 40-bit parallel adder capable of taking two binary or decimal inputs and producing the binary or decimal sum of these numbers as a result.

Shifter

The E-Unit shifter can perform a right or left shift of 0-63 bit positions in one machine cycle. The right or left shift is controlled by selectively rotating or not rotating the inputs and outputs. A right shifter can perform left shifts if the inputs and output are both rotated about their center axis.

Byte Mover

The 8-bit parallel mover is used to execute a subset of SS-type instructions and for exponent arithmetic in floating-point instructions. The byte mover is capable of two major functions. The first is taking two binary inputs and producing their binary sum as output. The second is the byte mover function. In this second role, the byte mover can assemble or disassemble digits to execute such instructions as PACK, UNPACK, MOVE ZONES, MOVE NUMERICS and EDIT.

DATA FLOW CONTROL

The E-Unit data flow consists basically of a group of registers, a group of functional components, and in-gates which interface the registers to the functional components. One cycle within the E-Unit consists of five logic levels and two latch levels, which are normally allocated as shown in Figure 5.3.

This data flow organization dictates two main decision points within one cycle. These are the in-gate level (0) and the register levels (5-6), with their clock and data select. Therefore, one group of control points is at the first level of the cycle, and another group is at the last level of the cycle. There are approximately twice as many in-gates as register gates to be controlled.

There are also a number of control signals required by the functional components. These are not nearly as numerous as the in-gates or registers. For the most part, these controls are required in the functional components either with the input data or a level later. The fact that more than two-thirds of all the control signals used on a given cycle are required during the first levels means that most of the control signals for that cycle will be generated and latched during the preceding cycle.

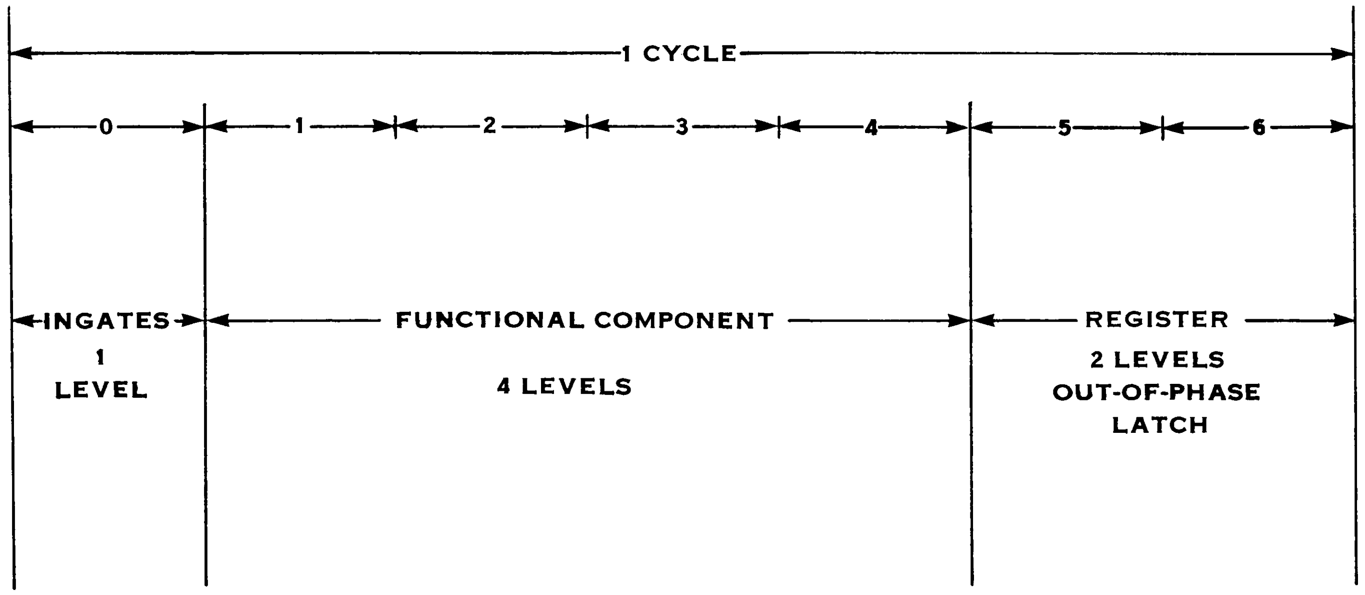


Figure 5.3 E-Unit Level Assignment

The E-Unit basically is concerned with controlling the execution of a single instruction. This execution can involve several cycles and several functional components. Most of these execution cycles will not be data dependent and hence will follow one another systematically until the required number of cycles has been completed.

6. INPUT/OUTPUT CHANNELS

The AMDAHL 470/6 System provides up to 16 channels for attaching input/output devices to the system. Any device compatible with the System 360/370 input/output interfaces listed below can be 8attached to the 470/6 System.

Channel data paths can be subdivided into separately-programmable subchannels. A subchannel is the facility that executes channel programs and is necessary to sustain an operation on an I/O device. One or more subchannels share the data transfer facility of a channel. The AMDAHL 470/6 configuration permits 1024 subchannels to be assigned in groups of 64, 128, and 256 per channel.

Three types of channels are available: byte-multiplexer channels, block-multiplexer channels, and selector channels. There are no design restrictions on the way channel types are assigned.

The functions of these channels and their subchannels are controlled by the channel unit. Figure 1.1 illustrates the relationship of this unit to the rest of the 470/6 System's logical structure.

CHANNEL UNIT (C-UNIT)

The components of the channel unit and the data flow between these components are shown in Figure 6.1.

Central Channel Unit

All data transferred between the S-Unit/I-Unit and C-Unit buffers or I/O devices must pass through the central channel unit. This unit is composed totally of LSI technology.

When an I/O instruction is encountered in the I-Unit's instruction pipeline, it is decoded and sent to the central channel unit for action. The I-Unit also transfers the necessary mask information for controlling I/O interruptions.

The priority of access to the S-Unit's high-speed buffer can be controlled by the C-Unit, allowing peremptory channel access to the buffer only when necessary. If the C-Unit is busy, it will have higher priority than CPU operand or instruction fetches. If the C-Unit is not busy, it can reduce its priority below these CPU operations. The C-Unit always takes precedence over a buffer prefetch.

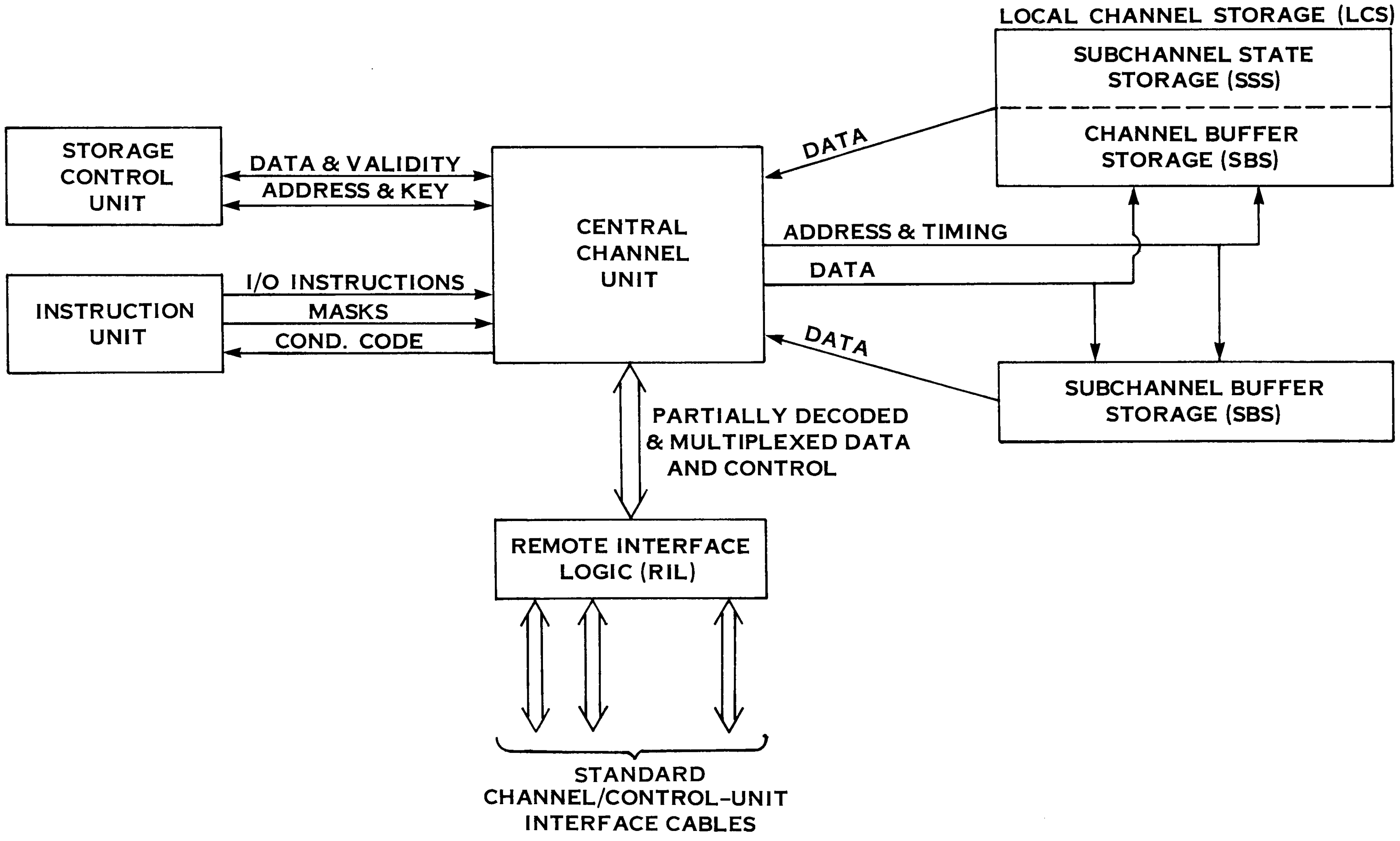

LOCAL CHANNEL STORAGE (LCS)
SUBCHANNEL STATE STORAGE (SSS)
CHANNEL BUFFER STORAGE (SBS)
STORAGE CONTROL UNIT
DATA & VALIDITY
ADDRESS & KEY
CENTRAL CHANNEL UNIT
DATA
ADDRESS & TIMING
DATA
INSTRUCTION UNIT
I/O INSTRUCTIONS
MASKS
COND. CODE
DATA
SUBCHANNEL BUFFER STORAGE (SBS)
PARTIALLY DECODED & MULTIPLEXED DATA AND CONTROL
REMOTE INTERFACE LOGIC (RIL)
STANDARD CHANNEL/CONTROL-UNIT INTERFACE CABLES

Figure 6.1 Channel-Unit Components

The I/O instructions recognized by the AMDAHL 470/6 System are:

- HALT DEVICE (HDV)
- HALT I/O (HIO)
- START I/O (SIO)
- START I/O FAST RELEASE (SIOF)
- STORE CHANNEL ID (STIDC)
- TEST CHANNEL (TCH)
- TEST I/O(TIO)

When it receives either an SIO or SIOF instruction, the central channel unit provides the S-Unit with the address and key information (contained in the channel address word—CAW) needed to fetch the appropriate channel command word (CCW). This address marks the beginning of the channel program. The CCW is decoded and the I/O function initiated by the central channel unit.

When the input/output instruction has been executed, the central channel unit is responsible for returning a condition code to the I-Unit. The condition code settings for each I/O instruction are shown in Figure 6.2.

Programming errors and equipment malfunctions usually set code 1; when this occurs the channel status word (CSW) is saved at main storage location 64 by the central channel unit. See the chapter "Input/Output Operations" in the document *IBM S/370 Principles of Operation* (GA22-7000) for further details.

Local Channel Storage

The local channel storage component of the C-Unit consists of two elements: channel buffer storage and subchannel state storage. Channel buffer storage contains 32 bytes of control words and 32 bytes of data buffer.

The subchannel state storage area contains one digit (half a byte) for each of the potential 1024 subchannels. Each byte holds subchannel status information needed for instruction response and interruption preparation.

Four priority levels are assigned to channel buffer storage, allowing the central channel unit to allocate control dynamically. High priority is assigned to an input channel as its channel buffer becomes 3/4 filled; the reverse is true for output channels. The highest intrachannel priority is assigned to a channel in danger of an overrun. If the channel is not in a stress situation, however, highest priority is given to a CCW fetch.

CODE SETTINGS						
Conditions	*HDV*	*HIO*	*SIO S10F*	*TCH*	*TIO*	*STIDC*
Channel, Subchannel, and Device Available	1	1	0,1	0	0	0
Interruption Pending in Device	1	1	1	0	1	1
Device Working	1	1	1	0	1	1
Device Not Operational	3	3	3	0	3	3
Interruption Pending in Subchannel						
For Addressed Device	0	0	2	0	1	2
For Another Device	0	0	2	0	2	2
Subchannel Working						
With Addressed Device	1*	1*	2	0	2	2
With Another Device	0	1*	2	0	2	2
Subchannel Not Operational	3	3	3	0	3	3
Interruption Pending in Channel	SA	SA	SA	1	SA	2
Channel Working						
With Addressed Device	IS	2	2	2	2	2
With Another Device	SS	2	2	2	2	2
Channel Not Operational	3	3	3	3	3	3

NUMERICAL CODE INTERPRETATION

Operation	0	1	2	3
Halt Device	Busy	CSW Stored	Working	Not Operational
Halt I/O	Pending Interruption	CSW Stored	Burst Operation Terminated	Not Operational
Start I/O	Started	CSW Stored	Busy	Not Operational
Start I/O Fast Release	Started	CSW Stored	Busy	Not Operational
Store Channel ID	ID Stored	CSW Stored	Busy; No Store	Not Operational
Test Channel	Available	Pending Interruption	Burst Mode	Not Operational
Test I/O	Available	CSW Stored	Busy	Not Operational

NON-NUMERICAL CODE INTERPRETATION

* If "device not operational" is returned when selecting the addressed device, CC=3.

IS Condition code depends on I/O interface sequence and channel type. If device has received termination signal, CC=1; otherwise, CC=2.

SA Same as corresponding "channel available" states (e.g., codes for pending interruption on channel with subchannel available and device working are the same as for the channel and subchannel available with device working state).

SS Code depends on subchannel state and device type. If subchannel is not operational, CC=2 or 3. If subchannel is available or working with addressed device, CC=2. Otherwise, CC=0 or 2.

Figure 6.2 Input/Output Condition Codes

The dynamic priority allocation systems for internal control and S-Unit access offer the following advantages over a system that uses fixed priorities:

- High-speed devices can be assigned to any channel without performance degradation;
- Channel interference with CPU accesses to the high-speed buffer is reduced, decreasing CPU access time.

Subchannel Buffer Storage

The subchannel buffer storage area contains infrequently referenced subchannel-control information. It consists of four words for each of the potential 1024 subchannels.

Remote Interface Logic

This component provides the interface between the central channel unit and the input/output devices' control units. One set of interface circuitry is provided for each I/O channel. The interface allows either a 1-byte or 2-byte data transfer path between the central channel unit and the I/O devices. The 2-byte interface feature can be attached to all channel types and each channel can use data-in/data-out control sequences. However, the channels interfacing with low-speed devices are usually limited by those devices' control units to the 1-byte interface without data-in/data-out.

CHANNEL TYPES

Byte-Multiplexer Channel

The byte-multiplexer channel allows a wide range of low-to-high-speed input/output devices to be attached to the AMDAHL 470/6 System. Up to 256 subchannels can be supported on this type of channel.

The byte-multiplexer channel operates in one of two modes: *byte-multiplex* mode for lower data rates, and *burst* mode for high-speed data transfer.

In byte-multiplex mode, the channel can control many low-speed devices concurrently (card readers, printers, etc.). Data are transferred between the devices and the byte-multiplexer channel on demand.

In burst mode, the control and data handling facilities of a channel are dedicated to a high-speed device (magnetic tape, disk, drum) for the duration of the data transfer.

The maximum data rate that can be sustained on a byte-multiplexer channel depends on the features supported by the control unit and the number of bytes transferred at each connection. The transfer rates vary from single-byte transfers at up to 135,000 bytes per second to very long or burst-mode transfers at up to 4,000,000 bytes per second (see Figure 1.2). Transfer rates also vary in relation to cable length.

Selector Channel

The selector channel can transmit up to 2,000,000 bytes per second using the 1-byte interface and up to 4,000,000 bytes per second with the 2-byte interface. This model operates solely in burst mode, handling one I/O device at a time. Normally this will be a high-speed device to take advantage of the selector channel's full capabilities. Transfer rates will vary in relation to cable length.

Block-Multiplexer Channel

The block-multiplexer channel operates in one of two modes: *selector* or *block-multiplex.* In the first mode, the channel effectively functions like a selector channel. The second mode, the block-multiplex mode, permits the concurrent control of several I/O devices, as with byte-multiplexer channels. Byte and block multiplexers differ mainly in that block-multiplexer channels handle faster devices and transfer larger blocks of data per transmission. For example, a block-multiplexer channel can control several Model 3330 Disk Units concurrently, combining the best features of both byte-multiplexer and selector channels.

Up to 2,000,000 bytes per second can be transmitted on a block-multiplexer channel using the 1-byte interface and up to 4,000,000 per second using the 2-byte interface. Transfer rates will vary in relation to cable length. As many as 256 subchannels can be supported.

If a multiplexed operation is initiated on a block-multiplexer channel by START I/O or START I/O FAST RELEASE, the multiplexing capability of the channel is determined from the block-multiplexing control bit (bit 0 of control register 0). When this bit is set to zero, multiplexing is inhibited; when it is set to one, multiplexing is allowed. The bit setting applies to the initiated operation until the related subchannel becomes available.

The block-multiplexing control bit is disregarded for all I/O instructions other than START I/O and START I/O FAST RELEASE. In particular, TEST I/O can clear an interruption pending in a subchannel and HALT DEVICE can cause an operation (if any) in the addressed subchannel to be terminated, even with multiplexing inhibited.

One or more operations allowing multiplexing and an operation inhibiting multiplexing can be executed simultaneously by a block-multiplexer channel. To ensure complete compatibility with selector channel operation, all operational subchannels on a block-multiplexer channel should be available or have multiplexing inhibited when selector mode is begun on that channel. All subsequent operations should then be initiated with block multiplexing inhibited.

EXTENDED CHANNEL MASKING

Control register 2 allows for selective masking of each I/O channel. Bit positions 0-31 of the control register can mask up to 32 channels. The maximum number currently supported by AMDAHL software is 16.

Interruptions from channels 0-5 are controlled by the channel masks in bit positions 0-5 of the PSW. The corresponding bits in the control register do not participate in interruption control.

Interruptions from channels numbered 6 or higher are controlled by the I/O mask bit (PSW bit 6) in conjunction with the corresponding mask bit in control register 2. Both bits must be set to one for an interruption to occur.

A system reset sets all channel mask bits to one.

COMMAND RETRY

The command retry feature causes a failing channel command to be retried automatically as often as the system controller requests. No I/O interruption is generated. On the AMDAHL 470/6 System, all channels support hardware command retry in addition to the software retry supported by OS/360.

Command retry is initiated by a control unit with a unique combination of status bits. The control unit may be trying to recover from a transient error or retrying the command because the previous state of the control unit or device prevented command execution.

7. SYSTEM CONSOLE

The system console provides operator control of the AMDAHL 470/6 operating system and hardware, plus extensive maintenance capabilities for the AMDAHL field engineer (FE). It contains its own computer and software, enabling a failing main computer to be diagnosed and controlled by a properly functioning console processor. Latches in the 470/6 CPU can be read and the machine's state made known to the field engineer.

The console components include the console processor with its own 16K, 16-bit word memory; two interfaces with the CPU; disk memory; a magnetic tape cassette; operator control panel; CRT display unit; communications modem; and keyboard (see Figures 7.1 and 7.2).

The two interfaces allow the system console three distinct modes of operation. Using the "channel interface," the console is functionally equivalent to an IBM CRT-type console from an operator's viewpoint. This is called device support mode. Using its own "computer-to-console interface" (CCI), it can operate in either system control mode or maintenance mode.

DEVICE SUPPORT MODE

In this mode, the console can perform all the OS-related operator functions of an IBM graphic operator's device. Specifically, the implementation of the system console causes it to imitate exactly an IBM 3066 (the S/370, Model l65, Graphic Console). The 3066 channel commands to which the 470/6 console responds are as follows:

Command	*Function*
Test I/O	This command tests the availability of an I/O device; the device or control unit returns its pending status.
Set Buffer Address	This control command specifies the address to which the next line of data will be read or written.
Write	This command writes a line or lines of data from the channel to the device. The starting line address is specified by the last Set Buffer Address command.
Set Cursor	This command specifies a new address with x and y coordinates where the cursor will be displayed on the CRT. It also unlocks the keyboard.

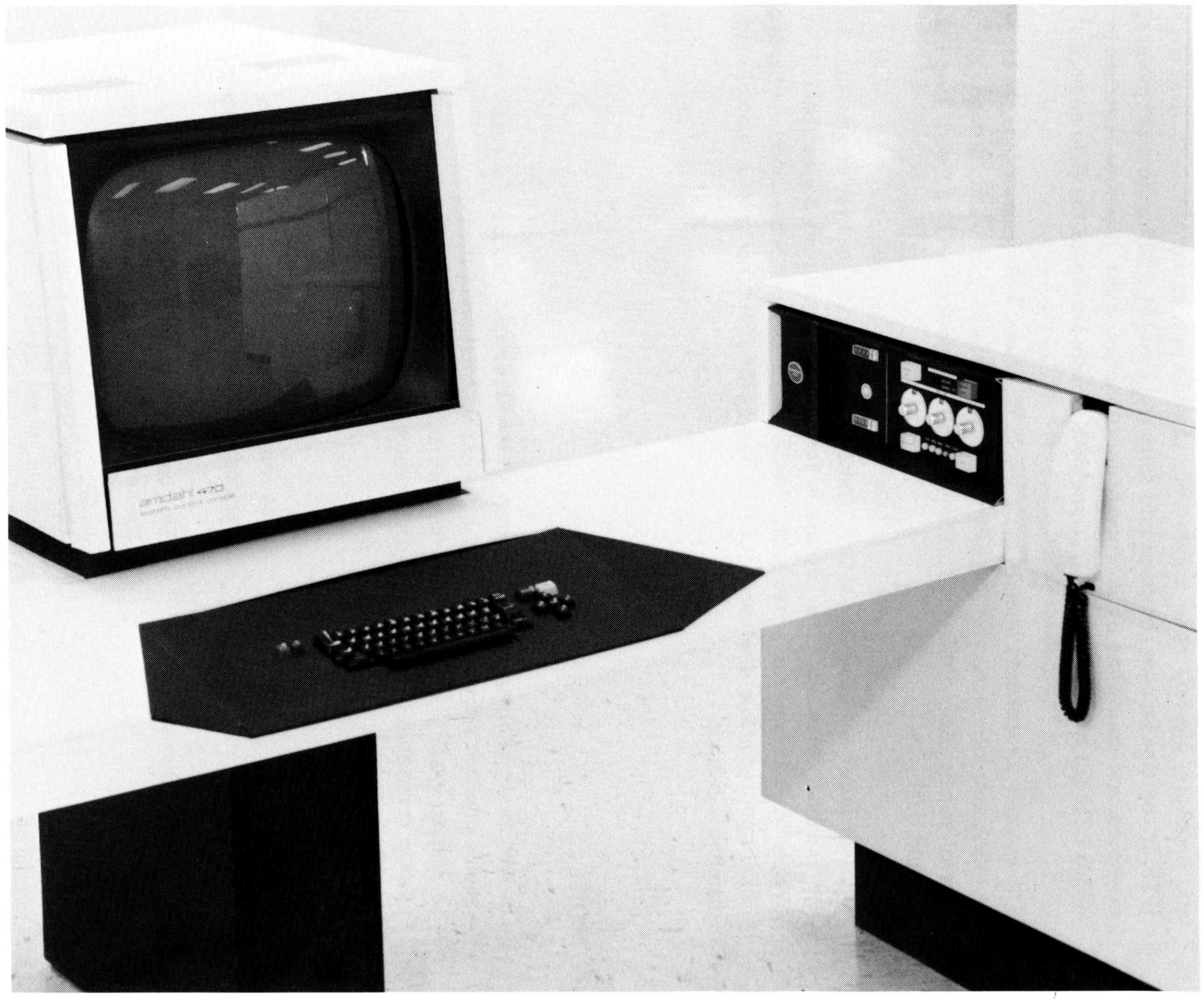


Figure 7.1 AMDAHL System Console

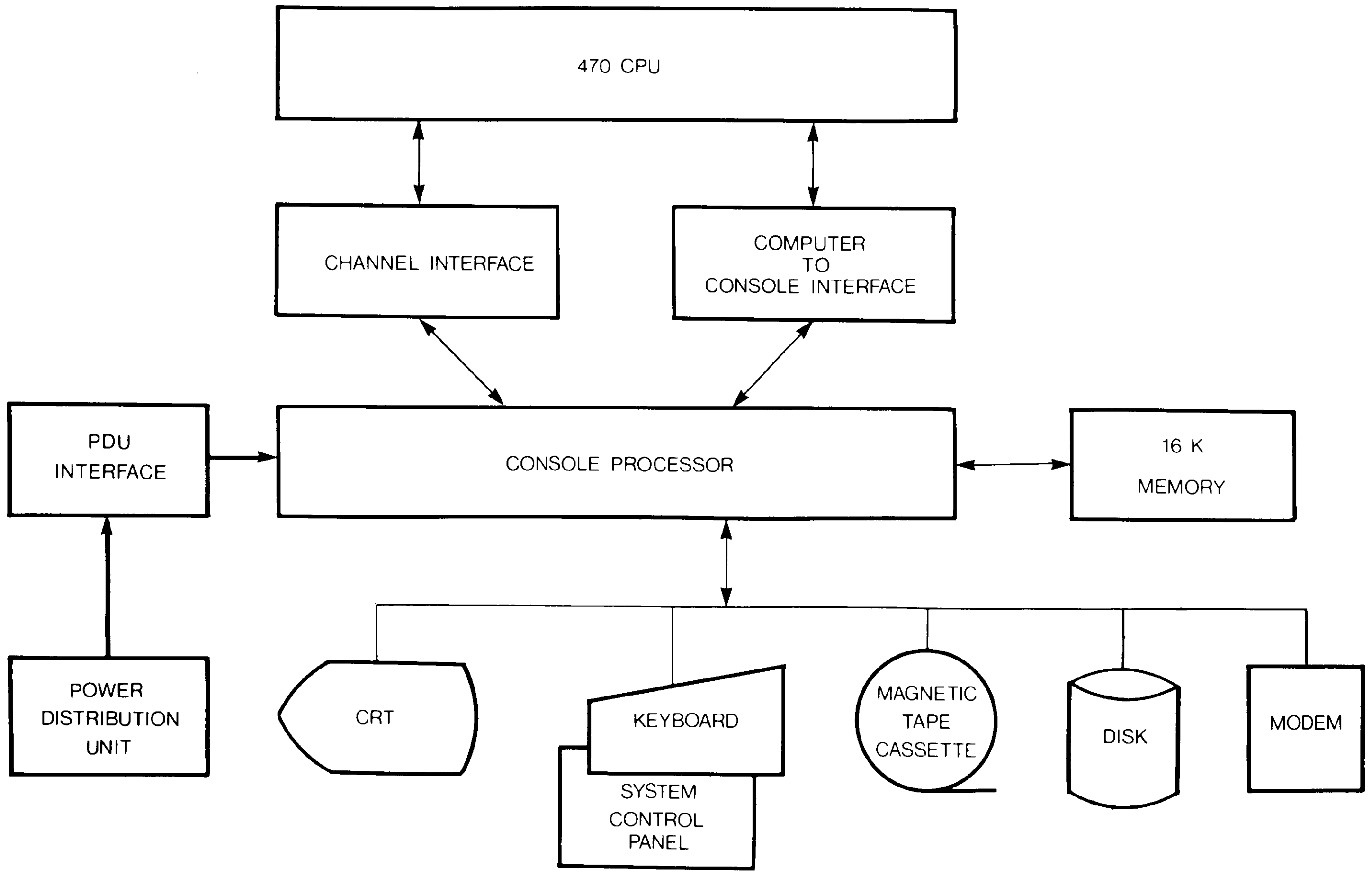


Figure 7.2 System Console Components

Read	This operation transfers data from the device buffer to the channel. The line address is specified by the last Set Buffer Address command.
Read Manual Input	OS responds to attention status from the operator console by issuing a Read Manual Input command. The command causes the device to send back three bytes of data containing the x and y cursor address and a status byte. The status byte indicates the reason for the attention interruption.
Set Audible Alarm	This command activates the audible alarm and illuminates the keyboard "alarm" key.
Lock Keyboard	This operation functionally disables the console keyboard for further input.
Erase	This command causes the following actions: • The screen is blanked; a space code is stored in each buffer location; • The alarm is turned off; • The buffer address is reset to zero; • The cursor x and y coordinates are set to zero; • The cursor display is inhibited; • The keyboard is locked.
Sense	The sense information for the device is returned to the channel.
No Op	The device returns an immediate device-end/channel-end to this command.

SYSTEM CONTROL MODE

The previous section described the commands used for operating system channel functions. This section describes control panel and keyboard functions performed in system control mode. System control mode applies to a situation where:

- The system has been stopped and the contents of the high-speed buffer, PSW, general-purpose or floating-point registers are displayed or altered via the CRT/Keyboard;

- The system has been powered on; initial program loading (IPL) is being performed;
- A reset function is required.

The operator in this mode can be either the console operator or the AMDAHL field engineer.

System Control Panel

One design principle of the system console is to do away with the mass of toggles and display lights normally found on control panels. Most console output functions appear as visual, formatted displays on the CRT; most input functions are performed using the console keyboard. The system control panel contains only the emergency-stop pull handle, and the controls needed for power-on functions, initial program loading, status indication, TOD clock control, and metering (Figure 7.3).

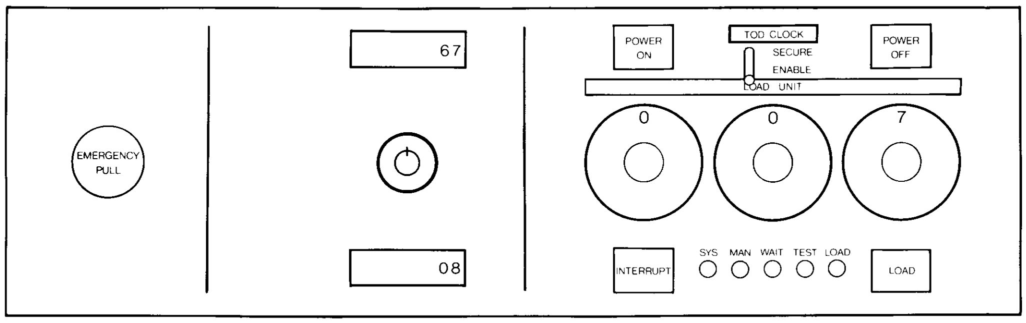


Figure 7.3 System Control Panel

Switches and Buttons

- Emergency Pull – Power off immediately, direct line to power control unit.
- IPL Panel – The switches in this section of the system console are:

  1) POWER ON
  2) POWER OFF
  3) INTERRUPT
  4) LOAD
  5) LOAD UNIT DIALS
  6) TOD CLOCK ENABLE

These switches cause the console, the 470/6 and the power distribution unit to perform certain operations. POWER ON brings power to the power distribution unit, powers on the console computer and its devices, causes a program load of the console processor, and powers on the 470/6 and all control units attached to it.

Pressing the POWER ON button causes the console's Power-On Initialization Program (POP) to monitor the 470/6 main-frame and control units' power-on sequence. If these operations are successful, POP initializes the 470/6's operating-state registers (OPSRs), main storage, high-speed buffer, control registers and channel buffers.

Pressing the LOAD button initiates the initial program load (IPL) operations. The functions of this program are:

1) Reinitialization of console processor;
2) System reset of the 470/6;
3) Initialization of OPSRs, control registers, and channel buffers;
4) Loading of device support programs into the console processor and initialization of these programs;
5) Control of IPL in the 470/6 (loading program into 470/6 from device addressed by LOAD UNIT dials);
6) Controlling LOAD indicator light.

See the description of the power distribution unit in Chapter 8 for the alternate power-on procedure.

Indicators

The five indicators on the IPL panel are WAIT, MANUAL, SYSTEM, TEST, and LOAD. WAIT, MANUAL and SYSTEM will be driven by the 470/6 hardware directly. The TEST and LOAD lights will be turned on and off by the console programs when those conditions arise.

Metering Assembly

The metering assembly contains the CPU meter, field engineering authorization keyswitch and FE meter. The CPU meter operates when the 470/6 CPU is running or the I/O system is active and FE maintenance mode is not authorized. The FE meter operates when the FE mode is authorized.

Console Keyboard

Keyboard Character Set

The system console uses a standard alphanumeric keyboard laid out as shown in Figure 7.4. The character set of the system console keyboard is listed below.

- Alphabetic Upper Case A→Z
- Numeric 0→9
- Special Symbols

=	Equal Sign
,	Comma
.	Period
$	Dollar Sign
/	Slash
@	Commercial At Sign
#	Pound Sign
+	Plus
<	Less Than
;	Semicolon
:	Colon
%	Percent Sign
>	Greater Than
*	Asterisk
(	Left Parenthesis
)	Right Parenthesis
|	Vertical Bar
¬	Not Sign
!	Exclamation Point
_	Underscore
-	Minus Sign, Hyphen
?	Question Mark
¢	Cent Sign
"	Quotation Mark
'	Apostrophe
&	Ampersand

- Space

Special Keys

In addition, the console keyboard contains certain special-function keys as follows:

- Start Key – issues a "start processing" or a single clock pulse command to the 470/6 CPU, depending on the state of the 470/6.
- Stop Key – Issues a "stop processing" command to the 470/6 CPU.
- Request Key – Console processor unlocks keyboard and signals READY or READ state for transmission of commands or data.
- Enter Key – Indicates end of message or deletion of the line depending upon the cursor location.
- Backspace Key – Backspaces CRT cursor one character from current cursor location and blanks out new cursor location.
- Cancel Key – Blanks out all the characters in the line being entered and repositions cursor to the beginning of the entry line.
- Cursor Up – Moves cursor up one line at a time; typematic key (when typematic key is held down, the function continues repeatedly).
- Cursor Down – Moves cursor down one line at a time; typematic key.
- Cursor Back – Moves cursor backwards one space at a time; typematic key.
- Cursor Forward – Moves cursor forward one space at a time; typematic key.
- Cursor Home – Moves cursor to home position (first position of entry, or bottom, line).
- Alarm – Resets audible alarm and extinguishes keyboard alarm light.

Keyboard Commands

The keyboard commands available to the operator in system control mode are as follows:

Command		*Function*
Reset	S-unit	Reset S-unit
	C-unit	Reset channels, control units and C-unit.
	I-unit	Reset I-unit and E-unit.
	Check	Reset all checks.
	System	All of the above.
Restart		Store current PSW in locations 8-15, load PSW from location 0 and start.
Rate	Process	Normal mode, full-speed execution
	Instruction n-times	Executes n instructions plus any allowable interruptions that become pending during the n instructions; n default is 1. The rate options are mutually exclusive; only one can be chosen.
Address Stop (Set)	Instruction Address Operand Fetch Operand Store C-unit Address Fetch C-unit Address Store All of the Above	Stop system when an equal comparison occurs. Any combination can be chosen for any one address.
Address Stop (Remove)		Removes all address stops set in the 470/6.
Check Control	Stop	Stop system in the event of a machine check without logout.
	Process	Normal mode, hardware logout and MCI. Used to reenable checks.
	Disable Soft	No interruption on soft machine checks.
	Disable Hard	No interruption on hard machine checks.
Interval Timer	Disable	Stop automatic hardware decrementing of the location 80 timer.
	Enable	Normal mode, automatic decrementing of the timer.
Display	High-Speed Buffer	Display 32 bytes of the HSB from a given address.
	PSW	Display the current PSW.

Command		Function
	GPR	Display 16 general-purpose registers.
	FPR	Display the four floating-point registers.
Store	High-Speed Buffer PSW GPR FPR	Alter the contents of the HSB, PSW, GPR, or FPR by the values given by the operator.

MAINTENANCE MODE

Maintenance mode refers to a situation where the AMDAHL field engineer is performing diagnostic tests. The advanced design of the 470/6 System permits maintenance operations to be performed under console processor control, even though the main computer is inoperable. The result is a very high level of system maintainability and availability.

Maintenance mode functions are available only to the field engineer. Authorization is enforced by the console processor programs.

RESETS

The keyboard commands available in system control and maintenance modes allow the user to initiate various reset operations externally. There are four basic reset categories: CPU, system, system clear, and power on.

CPU Reset

A CPU reset clears all machine-check, supervisor-call, program and external interruption conditions (see Chapter 9). The operating state registers, S-Unit and I-Unit are initialized and the system enters the stopped state.

When processing resumes, any instruction operands or pending store operations pre-fetched before the CPU reset are not acted upon unless they are fetched again after the CPU reset. The state of the C-Unit and attached control units is not affected and I/O interruption conditions are not cleared, provided the CPU is initially in the stopped state. When the CPU reset function is initiated while the CPU is executing an I/O instruction, taking an I/O interruption, or performing the initial program loading function, the resultant state of the associated channel, subchannel and I/O device is unpredictable.

The PSW, control registers, general-purpose registers, floating-point registers, storage keys, contents of main storage and time-of-day clock are unaffected.

System Reset

When a system reset is performed, the CPU and channels are reset; parity on the PSW, control registers, general-purpose registers and floating-point registers is validated; and the store status function is executed. The system reset is usually performed to clear equipment check conditions with minimal disturbance to the machine state, as when operations are to be resumed or system analysis is desired.

The channel reset sequence involves issuing a C-Unit reset, loading the shifting channel state (SCS) with a system reset, and clearing the local channel storage and subchannel buffer storage areas. The channels are in the idle state following this sequence.

If the channel reset is initiated while data are being stored from the C-Unit to main storage, the main-storage address referenced by the S-Unit address register (in the C-Unit) may be loaded with invalid data.

System-Clear Reset

A system reset is performed as part of the system-clear reset function. In addition the PSW, general-purpose and floating-point registers, main memory and storage-protection key are cleared. The control registers are set to their initial values (see Figure 9.2).

Power-On Reset

The power-on reset performs all the functions of the system-clear reset and additionally clears the time-of-day clock. After the power-on reset is completed, the system is in the stopped state.

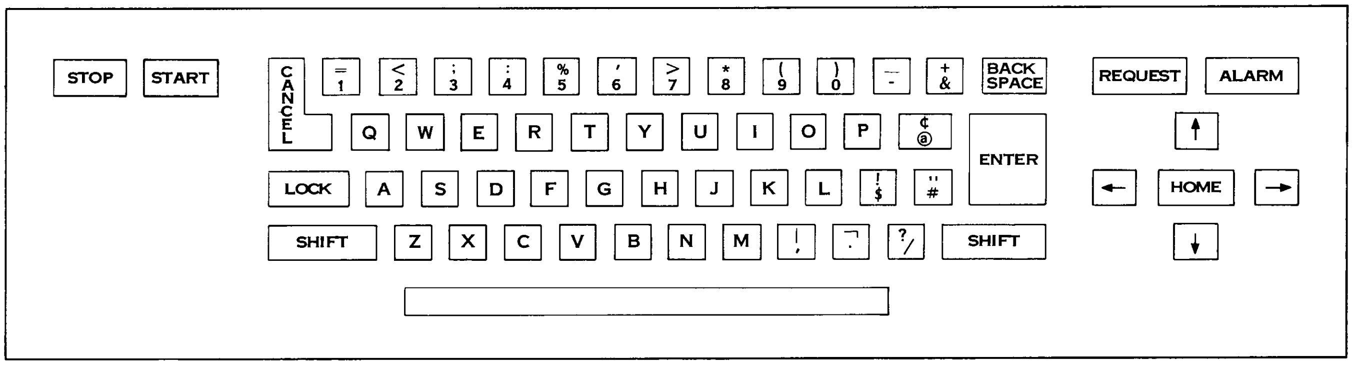


Figure 7.4 Console Keyboard Layout

8. AMDAHL 470/6 PHYSICAL COMPLEX AND PDU

PHYSICAL COMPLEX

Most of the components of the AMDAHL 470/6 System are contained in a single physical unit (Figure 8.1). This physical complex includes:

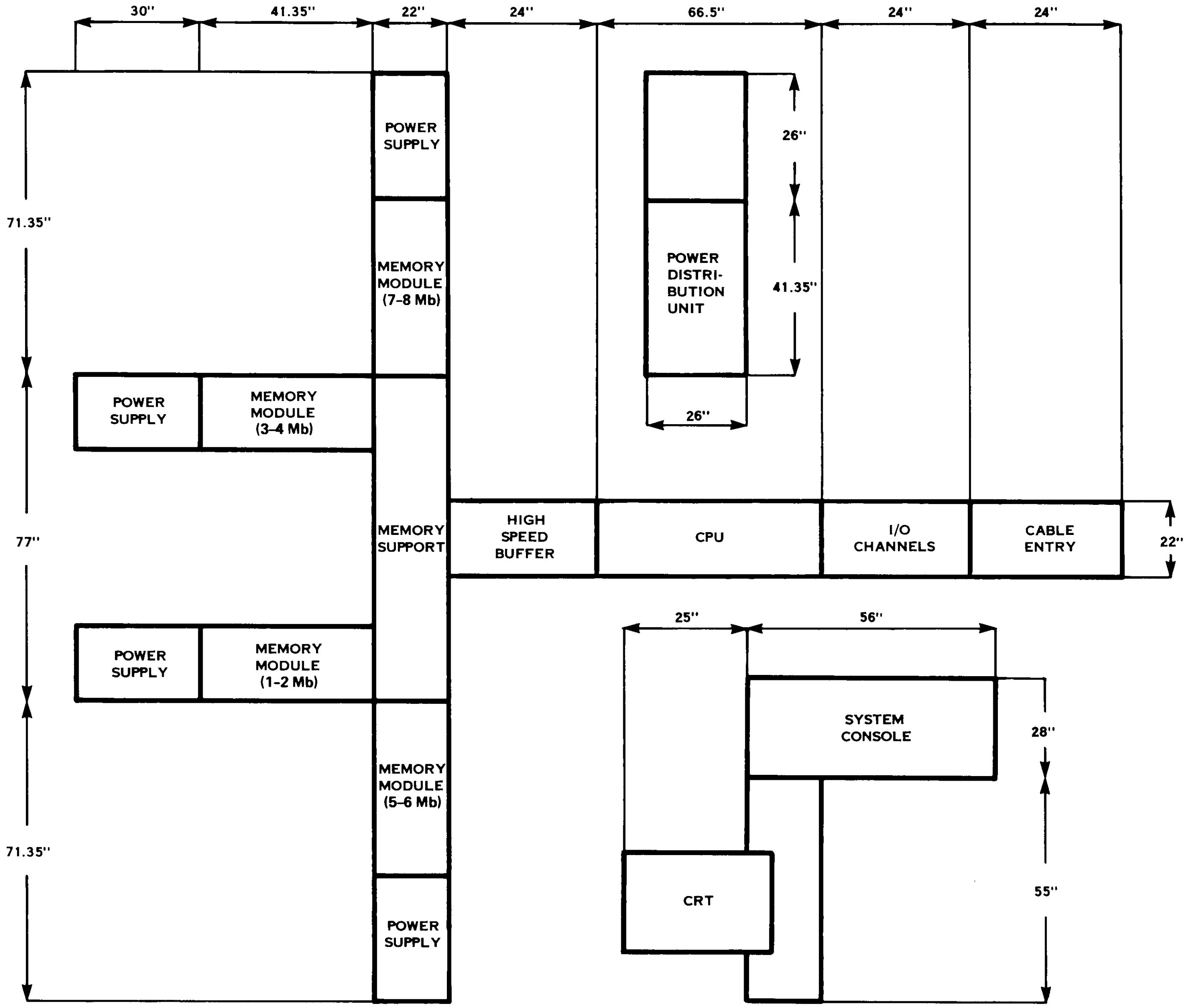
- The central processing unit;
- The 1M-8M main storage configurations;
- Up to 16 channels in any configuration.

System components that are physically separate from this complex include:

- The system console, which includes a control panel, CRT display, and integrated operator keyboard;
- Power distribution unit;
- A customer-supplied motor-generator set to supply 400-cycle power and power-system-isolation protection.

The main storage, CPU, channel and console components are described in Chapters 2-7. The power distribution unit is described in this chapter.

Specifications are detailed in the *AMDAHL 470/6 System Physical Planning Manual* (G414).



NOTE: Dimensions shown do not include covers and are not drawn to scale.

Figure 8.1 AMDAHL 470 Physical Complex (8M Configuration)

POWER DISTRIBUTION UNIT

The power distribution unit (PDU) performs the following functions:

- Controls and distributes 60 Hz and 415 Hz, three-phase AC power (supplied by the customer) to the console and main-frame units;
- Controls power-on/power-off sequences of the main-frame and control units;
- Monitors the power and thermal status of the main-frame units;
- Provides emergency power off interface to all units in the system.

Up to 16 AMDAHL main-frame units and 112 control units (standard emergency power-off interfaces) can be attached to the PDU. Main-frame units are designated as follows:

01	CPU
02	Buffer
03	Channel
04	Main Storage 1
05	Main Storage 2
06	Main Storage 3
07	Main Storage 4

AC Power Distribution

The PDU controls and distributes 60 Hz, three-phase, 208 volts to the console and main-frame units to operate fans and blowers and 60 Hz, one-phase, 115 volts to the console and main-frame convenience outlets. It also controls and distributes 415 Hz, three-phase, 208 volts to the DC supplies in the main-frame units and 415 Hz, single-phase, 208 volts to the console disk. The motor-generator set that supplies the 415 Hz power is owned, installed and serviced by the customer.

System Power Sequence Control

Power-On Sequence

System power can be turned on from the console or the PDU, depending on the setting of the test/normal switch on the PDU test panel. With this switch at "normal," power is turned on by pressing the system POWER ON pushbutton at the console. If the switch is set to "test" position, power is turned on by pressing the system POWER ON pushbutton on the PDU test panel. Both pushbuttons are backlighted red until the system power-on sequence is completed, then they change to white. The POWER ON lights at the console are powered and driven by the PDU.

Power-Off Sequence

System POWER OFF pushbuttons are located at the console and the PDU test panel. Pressing the POWER OFF pushbutton starts the power-off sequence in the reverse order of the power-on sequence.

At the start of the power-off sequence, the POWER ON pushbutton changes from white to red. After the power-off sequence is completed, the red indicator is turned off. The fans in the main-frame units are held on for five minutes after system power-off.

Power Status Monitoring

The PDU monitors the power and thermal status of the main-frame units.

An "alarm signal" is sent to the console when the 415 Hz source drops below normal voltage.

A "power check" signal, together with the unit address, is sent to the console if any one of the main-frame units is turned off or will not sequence on or off.

A "thermal check" signal, together with the unit address, is sent to the console if any one of the thermal switches is being activated in the main-frame units. DC power in the failing unit is turned off after five seconds, regardless of console masks.

A "margin voltage" signal, together with the unit address, is sent to the console if any one of the power supplies in the system is being margined or left at margin position.

Emergency Power Off

Activating the emergency power-off switch at the console or PDU opens the main contactors for the three-phase 415 Hz and three-phase 60 Hz to the system and single-phase 415 Hz and 60 Hz to the console.

The emergency power-off switch, once open, is latched open mechanically and can be reset only from behind the panel.

9. INTERRUPTION HANDLING

The interruption-handling facilities of the AMDAHL 470/6 System respond dynamically to hardware and software errors and certain nonerror conditions detected during system operation. When an interruption signal is encountered, control is switched to an interruption servicing program.

As soon as a program is interrupted, all current information about its status and a code identifying the cause of the interruption are stored in the program status word. This "old" PSW is then saved in a fixed low-memory location. A "new" PSW, designed to service the identified interruption, is fetched and becomes the controlling PSW. After the interruption has been serviced, the "old" PSW (status of the CPU before the interruption) may be restored.

Interruptive signals can come from several sources. They are honored in the following priority sequence:

- Hard machine check
- Supervisor call interruption
- Program interruption
- Soft machine check
- External interruptions
- Input/Output interruptions
- Restart interruption

Program, supervisor-call, external and restart interruptions are processed exactly as in S/370. Machine-check and channel-check handling routines are slightly different, however, to allow for unique error returns. These are differences normally required when moving to different models within the 370 line.

CLASSES OF INTERRUPTIONS

Machine-Check Interruptions

Machine-check conditions are signaled when the machine-checking circuits detect an error. A machine-check interruption (MCI) can be either "hard" or "soft," depending on its severity.

A hard MCI initiates an immediate termination of CPU activity accompanied by a logout of machine status. Expanded machine-check and processor-logout facilities in the AMDAHL 470/6 System aid in the rapid isolation and repair of these problems.

A soft MCI, on the other hand, is noncritical and is not acted upon until the current operation is completed. At the time of the interruption, error information is recorded in the logout area for analysis.

The system software includes Recovery Management Support (RMS) code that has the ability to distinguish between hard and soft machine-check interruptions, to assess program damage, and to terminate jobs (or not) on a selective basis.

Supervisor-Call Interruptions

A problem program passes control to the supervisor portion of the operating system's control program by issuing a supervisor-call instruction.

Program Interruptions

Program interruptions are the result of various kinds of programming errors and exceptions.

External Interruptions

An external interruption can be caused by the operator pressing the interrupt key on the system control panel, by the interval timer (an internal clocking device) becoming negative, or when an appropriate signal is generated by the direct control feature.

Input/Output Interruptions

Input/output interruptions result when an I/O unit has terminated its operation or needs attention. The identification of the device and channel causing the interruption is stored in the "old" PSW. Device and channel status is stored in a channel status word (CSW).

Restart Interruptions

A restart interruption occurs when the operator enters the restart command through the system console.

INTERRUPTION CONTROLS

Most interruptions can be either suppressed or delayed, depending on the class of interruption. This is partially accomplished by setting certain bits in the PSW to zero (Figure 9.1). These bits act as masks corresponding to specific input/output, external, machine-check, and program interruptions.

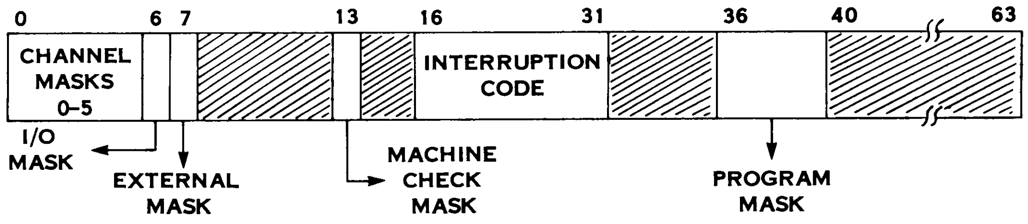


Figure 9.1 PSW Interruption Masks

The AMDAHL 470/6 System structure also provides for up to 16 control registers of 32-bits each, allowing extended external, I/O, and MCI masking functions (Figure 9.2). These registers are available to, and managed by, the operating system and in this manner control the state of the machine.

Two instructions are available for loading and storing control registers. LOAD CONTROL is used to load control data from main storage into the control registers. STORE CONTROL is used to transfer data from control registers into main storage. Both instructions are privileged.

Ctl Reg	Bits	Field	Function	Value After Reset
0	0	Block-Multiplex Mode	Block-Multiplexing Indicator	0
0	24	Timer Mask	Extended	1
	25	Interrupt-Key Mask	External	1
	26	External-Signal Mask	Masking	1
2	6-31	Channel Masks	Extended I/O Masking	1
14	0	Hard-Stop Mode	Machine-Check	1
	2	I/O Extended Logout	Handling	0
	4	Recovery Report Mask		0
	6	External-Damage Report Mask		1

Figure 9.2 Control-Register Bit Assignments

MACHINE-CHECK HANDLING

Data in storage and registers are normally assumed to have correct checking codes. A machine-check condition occurs when instructions or data with an invalid checking code are fetched (whether or not the data are used), when an arithmetic function is performed improperly, or when invalid parity on the protection key or the storage key leaves the system unable to determine whether the particular storage reference is protected.

Once valid data are placed in storage or registers, a machine-check condition is caused only by a machine malfunction and never by logically invalid data or instructions. Specification of an unavailable system component such as a storage unit, channel, or I/O device does not cause an equipment malfunction. Instead, the appropriate program or I/O interruption results or a condition code is set.

A malfunction detected by the S-Unit during an I/O operation generates a machine-check condition. When the I/O operation causes fetching of a channel command word (CCW) or data, information identifying the malfunction is also included in the channel status (whether the CCW or data are actually used or not). The channel does not set status bits for malfunctions detected in the S-Unit on data stored by the channel. The channel does check such data at its S-Unit interface, however, and forces good parity.

Equipment malfunctions detected in channels are recognized in the channel status and are not treated as machine-check conditions. Error indication is not subject to the setting of PSW bit 13, the machine-check mask.

Machine-Check Conditions

Machine-check interruptions can be either soft or hard. Several machine-check "conditions" can be identified within these two categories.

Soft machine-check conditions never result in termination of the current instruction or cause interruptions to be lost. These include:

- System-Recovery Condition: an equipment malfunction that is successfully corrected or circumvented;

- Interval-Timer Damage Condition: a malfunction associated with the automatic updating of location 80 in memory;

- Time-of-Day-Clock Damage Condition: a malfunction in the operation of the time-of-day clock;

  NOTE: The system software currently treats this condition as a hard check and brings the system down.

- External-Damage Condition: a storage unit malfunction associated with a channel or prefetch operation.

Hard machine-check conditions may result in termination of the current instruction or cause interruptions to be lost. These include:

- **Instruction-Processing-Damage: a CPU malfunction asso**ciated with the instruction indicated by the machine-check old PSW;

- System-Damage Condition: a malfunction that cannot be pinpointed by a specific process.

Machine-Check Logout

A machine-check logout involves storing information related to a machine error. In general, an MCI causes the PSW active at the time of the interruption to be stored as the old PSW at location 48. Additional MCI information is stored in the fixed-logout area, the machine-check interruption code (MCIC), and the MCI extended information area.

It should be noted that the I-Unit can also perform a synchronous extended logout under console control. I-Unit operation is halted and a scanout on the console display is performed, after which operation can be resumed.

Logout Areas

As is shown in Figure 9.3, the MCIC occupies storage locations 232-239. The contents of these locations are described in detail later in this section.

The 96-byte area starting at main storage location 256 is called the fixed-logout area. The AMDAHL 470 currently uses the first 12 bytes of this area. Locations 256-259 contain the failing storage address. The doubleword starting at location 260 contains information specifying where the error occurred and is called the region code. A bit-by-bit listing of the contents of the region code can be found in Figures 9.12 and 9.13 at the end of this chapter.

On all machine-check interruptions, the addressable registers are stored sequentially. These data are referred to as the MCI extended information. Floating-Point registers 0, 2, 4, and 6 are stored starting at location 352; general-purpose registers 0-15 start at location 384; and control registers 0-15 start at location 448. Locations assigned to unimplemented control registers are set to zero.

Each of the MCI extended information (register) fields has an MCIC validity bit associated with it. If for any reason the machine cannot save one of these fields or cannot store the field validly, the associated validity bit is set to zero. The same is true of the failing-storage-address area.

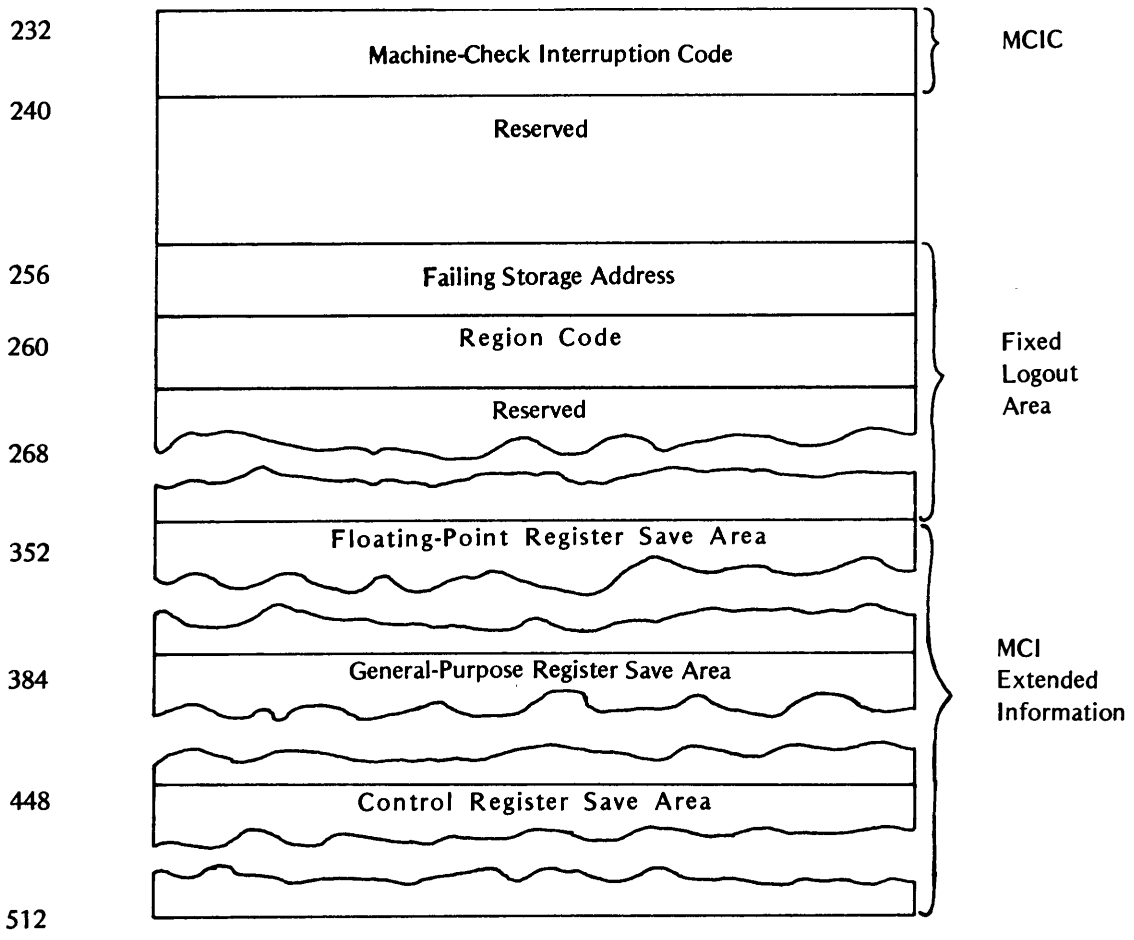

232
240
256
260
268
352
384
448
512
Machine-Check Interruption Code
Reserved
Failing Storage Address
Region Code
Reserved
Floating-Point Register Save Area
General-Purpose Register Save Area
Control Register Save Area
MCIC
Fixed Logout Area
MCI Extended Information

Figure 9.3 MCI Logout Locations

Machine-Check Control Register

Figure 9.4 identifies the machine-check mask bits in control register 14. These bits specify which conditions can allow an MCI and when a machine-check logout can occur. Bits 1, 3, 5, and 7-31 are reserved for future implementation.

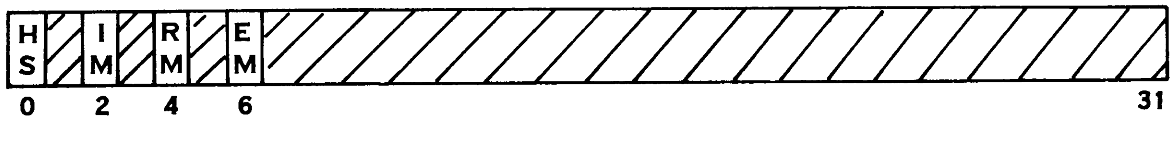


Figure 9.4 Control Register 14 Mask Bits

Hard Stop (HS), Bit 0

This mask bit controls the action taken when a hard machine-check condition occurs during an MCI or with PSW bit 13 set to zero. If the hard-stop bit is one, the CPU enters the check-stop state; if the hard-stop bit is zero, the machine tries to continue. The hard-stop bit is set to one by system reset. If any control register or the PSW is damaged, the hard-stop bit is assumed to be one.

When the machine is in the check-stop state, the system light is off, and the manual light is on. Instructions and interruptions are not executed, the timer is not updated, and channel operations are suspended. The stop key and start key are not operative, but the time-of-day clock remains operative. The CPU can be removed from the check-stop state by a system reset.

Input/Output Extended-Logout Mask (IM), Bit 2

When set to one, this bit permits the channel to log out into the I/O extended-logout area as part of an I/O interruption. When the mask is zero, such logouts cannot occur. This bit is set to zero by system reset.

Machine-Check Subclass Masks (RM & EM), Bits 4 and 6

These bits, in conjunction with PSW bit 13, control various machine-check subclass conditions. When the PSW bit is set to one and the subclass mask is one, the related condition initiates an MCI. If the subclass mask is zero, the corresponding RM or EM condition does not initiate an interruption, but remains pending. If another condition initiates an interruption, the pending condition is presented with the interrupting condition and all conditions presented are then cleared.

Recovery report mask (RM). Bit 4 determines whether an interruption will occur as a result of system recovery from an error. It is set to zero by a system reset.

External-damage report mask (EM). Bit 6 controls the timer-damage, time-of-day clock damage, and external-damage machine-check conditions. It is set to one by a system reset.

Figure 9.5 summarizes machine-check subclass-condition masking and resultant actions. See also the following discussion of the machine-check interruption code.

Machine-Check Interruption Code

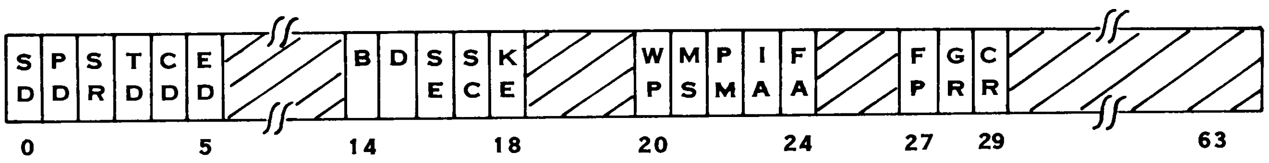
Another area where MCI-related information is stored is the doubleword field starting at main storage location 232. Data stored in this area are called the machine-check interruption code. The contents of the MCIC are summarized in Figure 9.6.

Subclass Condition		Mask	Action when CPU Disabled for Subclass Condition	
			Hard-Stop Bit = 0	Hard-Stop Bit = 1
SD	System Damage	PSW 13	P*	Check Stop
PD	Instruction-Processing Damage	PSW 13	P**	Check Stop
SR	System Recovery	13 or RM	P	P
TD	Timer Damage	13 or EM	P*	P*
CD	Time-of-Day-Clock Damage	13 or EM	P*	P*
ED	External Damage	13 or EM	P*	P*

Action Code Definition:

P	Indication held pending.
*	System integrity is undependable.
**	System integrity is undependable unless backup bit is on.

Figure 9.5 Machine-Check Subclass-Condition Masking



0-5	SUBCLASS CONDITIONS
14, 15	TIME OF INTERRUPTION
16-18	STORAGE ERROR TYPE
20-24, 27-29	VALIDITY
6-13, 19, 25-26, 30-63	NOT ASSIGNED, STORED AS ZERO

Figure 9.6 MCIC Fields

Subclass Conditions

Bits 0-5 identify the machine-check conditions causing the MCI. When a machine check occurs, at least one bit is stored as a one in the subclass field. When multiple errors occur, several bits are set to one. The items in this field are related directly to the subclass conditions shown in Figure 9.5.

System damage (SD). When MCIC bit 0 is set to one, interruptions may have been lost or damage has occurred that cannot be pinpointed to one or more of the less severe machine-check damage subclasses.

Errors classified as system damage include control register damage, PSW damage, unsuccessful retry of an interruption, CPU data damage in transfer from the S-Unit to main storage, or a CPU error not attributable to the instruction pointed to by the machine-check old PSW.

Instruction processing damage (PD). When bit 1 is set to one, a CPU malfunction has occurred in relation to the instruction associated with the machine-check old PSW. These are either unretryable or uncorrectable CPU errors, multiple-bit processor storage failures, or a storage-protect-key failure where the CPU is the requesting device.

System recovery (SR). When bit 2 is set to one, errors were detected but successfully corrected or circumvented with no loss of system integrity. A successfully retried CPU error or ECC function for any requesting device causes a system recovery condition.

Timer damage (TD). When bit 3 is set to one, a CPU or storage malfunction was detected while updating the interval timer at location 80 during an automatic timer update reference.

Time-of-day clock damage (CD). When bit 4 is set to one, the time-of-day clock has been damaged.

External damage (ED). When bit 5 is set to one, storage has been damaged during operations not directly associated with the CPU. Uncorrectable storage data errors detected during transfer to/from main storage for an I/O or instruction fetch request are sources of external damage. This bit is also set by damage to a channel, channel controller, switching unit, or other unit external to the CPU.

Time of Interruption

Bits 14 and 15 of the MCIC indicate when the MCI occurred, relative to the error.

Backup (B). If bit 14 is set to one, the point of interruption is before the point of error. This bit is set only when a retry attempt is unsuccessful.

Delayed (D). If bit 15 is set to one, some or all of the information stored as a result of the MCI was delayed. The delay resulted because the CPU was disabled for that type of interruption when the error was detected. Any information logged out subsequently will not be relevant to this MCI.

Storage Error Type

Bits 16-18 of the MCIC identify errors occurring in main storage or in a storage key as a result of internal or external storage requests. The failing-storage-address field (location 256), when valid, identifies the storage area in error. The portion of the system affected by the storage or protection error is indicated in the subclass field of the MCIC. Storage or protection errors involving prefetched or unused data are handled the same as errors on data in use.

Storage error uncorrected (SE). When bit 16 is set to one, a reference to storage caused or resulted in the detection of uncorrectable damaged data.

Storage error corrected (SC). When bit 17 is set to one, a reference to storage caused or resulted in the detection of an error that was corrected by the ECC hardware.

Key-in-storage error uncorrected (KE). When bit 18 is set to one, a reference to a key in storage caused or resulted in the detection of an uncorrectable error in the storage key. The keys are not checked for errors during storage references when the PSW or channel keys referring to storage are all zeros.

NOTE: The storage-error-type bits do not in themselves indicate that damage has occurred. The detected error might not have affected the result.

Validity Bits

Bits 20-24 and 27-29 of the MCIC are validity bits. Each bit indicates the validity of a particular field stored during the MCI-handling procedure. When a validity bit is set to one, it indicates that the error did not affect the original data and no error was detected when the data were stored.

PSW AMWP valid (WP). If bit 20 is set to one, bits 12-15 of the machine-check old PSW are valid.

PSW masks and key valid (MS). If bit 21 is set to one, PSW bits 0-11 of the machine-check old PSW (the system mask and key) are valid.

PSW program mask and condition code valid (PM). If bit 22 is set to one, the program mask and condition code in the machine-check old PSW are valid.

PSW instruction address valid (IA). If bit 23 is set to one, the instruction address in the old PSW (bits 40-63) accurately reflects the point in the instruction sequence at which the interruption occurred.

Failing storage address valid (FA). If bit 24 is set to one, the failing storage address (location 256) is valid (see notes below).

Floating-point registers valid (FP). If bit 27 is set to one, the contents of the floating-point-register save area (location 352) accurately reflect the status of these registers at the point of interruption.

General-purpose registers valid (GR). If bit 28 is set to one, the contents of the general-purpose-register save area (location 384) accurately reflect the status of these registers at the point of interruption.

Control registers valid (CR). If bit 29 is set to one, the contents of the control-register save area (location 448) accurately reflect the status of these registers at the point of interruption.

NOTES: The validity bits must be used in conjunction with the subclass and time-of-interruption bits to determine the extent of damage caused by the machine-check condition. The four PSW-valid bits and the three register-valid bits must be ones, and one of the following conditions must be true, to indicate that no damage has yet occurred to the system:

- The backup bit is set to zero, and all damage subclass bits are set to zero;

- The backup bit is one, and instruction-processing damage (PD, MCIC bit 1) is the only damage subclass bit set to one.

When a storage-error uncorrected (SE), uncorrectable storage-unit to main-storage, storage-error corrected (SC), or key-in-storage error uncorrected (KE) has been indicated in bits 16-18 of the MCIC, the failing storage address is stored (if available) in bits 8-31 of the word at location 256 (Figure 9.3). Bits 0-7 are set to zero. In the case of storage errors, the failing storage address may point to any address within the ECC block. If the KE bit is set to one, the failing storage address may point to any address within the 2048-byte block of storage related to the storage key in error. (When an error is detected in more than one location before the interruption, the failing storage address might point to any of the failing locations). If an error is detected, it is also possible that conditions exist preventing the failing storage address from being saved. This is indicated by resetting the failing storage address valid bit (MCIC bit 24).

Machine-Check Interruption Action

The actions taken by the CPU differ for the two classes of machine-check conditions:

- Those that are known to be soft (such as corrected single-bit storage errors, timer damage, time-of-day-clock damage, and storage errors resulting from I/O or prefetch operations);

- Those that might be hard (such as multiple-bit storage errors, storage-key parity errors, parity errors on data being moved between or within any of the units of the CPU, and errors in the results of arithmetic or logical operations).

Except where noted, it is assumed in this description that all mask bits are set to permit each of the actions (interruption, logout) to take place.

Soft Machine Checks

Soft machine checks cause an interruption just after the completion of the next instruction, including the associated SVC or program interruption if applicable. That is, it occurs at the same time as an external or I/O interruption would.

At the point of interruption, the MCI information and the old PSW (pointing to the instruction following the one just completed) are stored, and the new PSW is loaded from location 112.

Both PSW bit 13 and the associated subclass mask must be set to one for a soft machine-check condition to occur.

If a soft machine-check condition is noted while the CPU is disabled for a machine-check condition, the condition is held pending. Once a system recovery is signaled following the previous machine-check condition, the system-recovery condition will be held pending. The CPU never enters the check-stop state because of a soft machine-check condition.

> NOTE: A system recovery following a retry preserves the original fixed logout information. A second logout into this area does not occur.

Hard Machine Checks

When a potential hard machine check is detected, further updating of machine state (including registers and storage) is inhibited. This action is taken immediately, rather than at the end of an instruction. With the machine in this condition, there is an instruction that would be the next to complete execution (which, for the sake of illustration, is called I). In other words, I is the instruction furthest along in the instruction-unit pipeline. The instruction counter, which normally points to an instruction well beyond I, is rolled back to point to I. Note that the error could have been caused while executing any of the potential five instructions in the pipeline. This group of instructions includes I, but not the instruction before I.

Unretryable errors. If the error is not retryable (see Chapter 11), the following information is stored:

- General-purpose, floating-point, and control registers;
- Failing storage address, if available;
- Error register;
- Machine-check interruption code with:

–	System or processor damage bit	=	1
–	System-recovery bits	=	0
–	Backup Bit	=	0

After the information has been stored, the old PSW (with instruction address pointing to I) is stored and the new PSW loaded. Since the error is not retryable, the interruption occurs after I has probably begun to modify the machine state, but before I has been completed. Hence, the point of interruption occurs during the execution of I and I is terminated.

Retryable errors. If the error is retryable, an "asynchronous" fixed logout (error register and failing storage address, if available) is performed. The operation of the machine is then restarted by refilling the pipeline starting with instruction I.

> NOTE: If a fixed logout occurs as part of an MCI, it is called synchronous. If the logout occurs apart from an MCI, or if the MCI and logout operations are separated by instruction processing or retry, it is called asynchronous.

If an error occurs before I is completed this time, the error is assumed to be the same one and uncorrectable. The following information is stored:

- General-purpose, floating-point, and control registers;
- Machine-check interruption code with:

–	System or processor damage bit	=	1
–	System-recovery bit	=	0
–	Backup bit (unless I has passed its retry threshold)	=	1

After the information has been stored, the current PSW (with instruction address pointing to I) is stored and the new PSW loaded. If the error occurs before I is executed, I is suppressed and the backup bit is one. If the error occurs after I has begun to modify the machine state, the interruption point occurs while I is executing, I is terminated, and the backup bit is set to zero.

If no error occurs by the time I completes, the error is assumed to be corrected. The following information is stored:

- General-purpose, floating-point, and control registers;
- Machine-check interruption code with:

–	System and processor damage bit	=	0
–	System-recovery bit	=	1
–	Backup bit	=	0

The old PSW is then stored and the new PSW loaded. The interruption point occurs between I and the following instruction, and after any program or SVC interruptions associated with I have been taken.

Figure 9.7 summarizes the processing of potential hard machine checks.

An MCI caused by a hard machine-check condition can occur only when the machine-check mask, PSW bit 13, is set to one. When bit 13 is set to zero and a hard machine-check condition is signaled, subsequent action depends on the state of the hard-stop bit, bit 0 of control register 14. If this bit is zero, the machine-check condition is ignored and an attempt is made to complete the current instruction and to proceed normally. Operation in this mode may result in loss of system integrity. When the HS bit is one, processing stops immediately and the CPU enters the check-stop state. Hard machine-check conditions occurring while the HS bit is zero cannot cause the CPU to enter the check-stop state if the hard-stop bit is subsequently set to one.

Similarly, subsequent action depends on the state of the hard-stop bit if, during execution of the MCI procedure resulting from a previous hard machine-check condition, a second hard machine-check condition is detected. If the HS bit is set to one, the CPU enters the check-stop state: if the bit is zero, the CPU attempts to proceed. If a hard machine-check condition is detected during an MCI caused by a soft machine-check condition, the interruption becomes a system-damage report. Only one machine-check condition is held pending for each subclass, regardless of the number of conditions detected.

Every reasonable attempt is made to limit the side effects of hard machine-check conditions and the associated interruptions. Normally, I/O and external interruptions, as well as the progress of I/O data transfer and the updating of the internal timer, remain unaffected. The malfunction, however, may affect these activities. Furthermore, if the old PSW has bit 13 set to one, the MCI may terminate the switching of PSWs occurring because of another class of interruption.

MCIs can be initiated only by a condition in a subclass for which the CPU is enabled. Pending conditions in other subclasses might also be indicated in the same interruption, even though the CPU is not enabled for those subclasses. All indicated conditions are then cleared.

Machine-check conditions are handled identically in the running and wait states. In the wait state, a machine-check condition for which the CPU is enabled causes an immediate interruption.

Machine checks occurring during instruction-step-mode processing are handled in the same manner as in process mode (i.e., normal recovery, logout, and MCIs occur when allowed). Machine checks occurring during a manual operation, such as system reset, cause the CPU to enter the check-stop state.

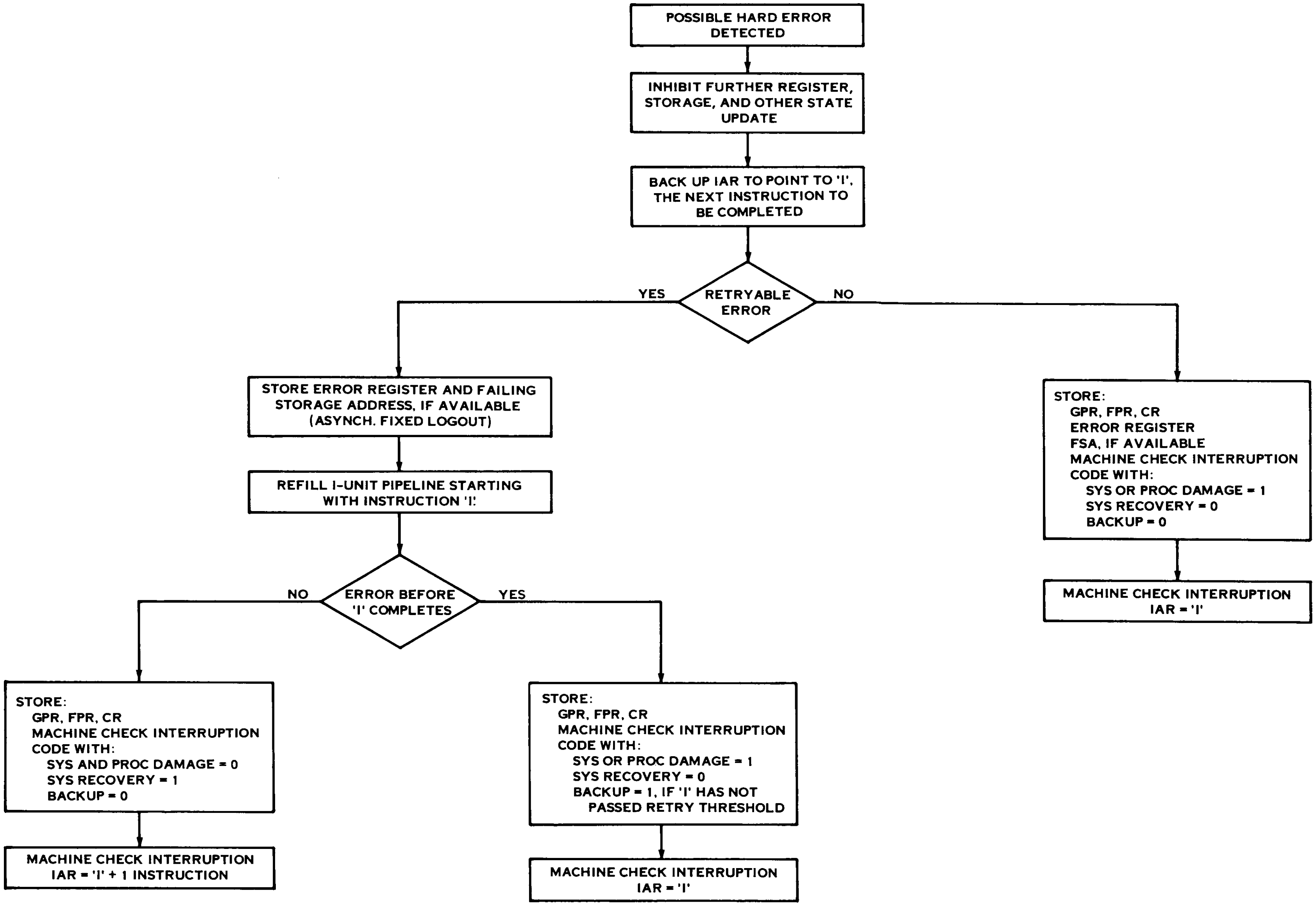

POSSIBLE HARD ERROR DETECTED
INHIBIT FURTHER REGISTER, STORAGE, AND OTHER STATE UPDATE
BACK UP IAR TO POINT TO 'I', THE NEXT INSTRUCTION TO BE COMPLETED
RETRYABLE ERROR
YES
NO
STORE ERROR REGISTER AND FAILING STORAGE ADDRESS, IF AVAILABLE (ASYNCH. FIXED LOGOUT)
REFILL I-UNIT PIPELINE STARTING WITH INSTRUCTION 'I'.
ERROR BEFORE 'I' COMPLETES
NO
YES
STORE: GPR, FPR, CR MACHINE CHECK INTERRUPTION CODE WITH: SYS AND PROC DAMAGE = 0 SYS RECOVERY = 1 BACKUP = 0
MACHINE CHECK INTERRUPTION IAR = 'I' + 1 INSTRUCTION
STORE: GPR, FPR, CR MACHINE CHECK INTERRUPTION CODE WITH: SYS OR PROC DAMAGE = 1 SYS RECOVERY = 0 BACKUP = 1, IF 'I' HAS NOT PASSED RETRY THRESHOLD
MACHINE CHECK INTERRUPTION IAR = 'I'
STORE: GPR, FPR, CR ERROR REGISTER FSA, IF AVAILABLE MACHINE CHECK INTERRUPTION CODE WITH: SYS OR PROC DAMAGE = 1 SYS RECOVERY = 0 BACKUP = 0
MACHINE CHECK INTERRUPTION IAR = 'I'

Figure 9.7 Handling of Potential Hard Machine Checks

CPU RECOVERY ACTION

Recovery from machine-detected malfunctions is attempted using two basic methods. Storage errors are handled by the ECC function described in Chapter 2. Processing errors are handled by the CPU retry function, which attempts to reexecute failing instructions.

Extensive checking facilities are contained within the CPU, including a parity check of instructions and data, and arithmetic checks on EAR and E-Unit functions. When an error is found, recovery is attempted by restarting an instruction automatically, provided it has not yet modified the machine state (e.g., main storage or a register that may be required during the retry). Many instructions modify the machine state only after all checks have been made. These instructions are completely retryable. (Chapter 11 lists nontryable instructions and the thresholds beyond which they become nonretryable).

If the error does not reoccur after the CPU retry, a "recovery" condition (i.e., a soft machine-check condition) is recognized. If the machine cannot execute the instruction after one retry, a hard machine-check condition is assumed.

CHANNEL-EQUIPMENT ERROR HANDLING

Channel-equipment errors are divided into three types:

- Channel-Data Check (CDC) – errors in data (as contrasted to control information) being transferred by the channel;
- Interface-Control Check (IFCC) – invalid signals on the I/O interface;
- Channel-Control Check (CCC) – any malfunction affecting channel controls. IFCC and CCC will not be indicated simultaneously. If both apply, only the channel control check is indicated.

When an error affecting the operation of a single channel is detected, it is reported only on that channel. (Example: parity error on a fetch from C-Unit internal storage). If an error can affect the operation of more than one channel, it is reported on each channel that can be affected. (Example: error in internal C-Unit channel synchronization). Note that this kind of error differs from a single failure resulting in errors associated with several channels at different times.

When an error is detected that the channel cannot report using the I/O interruption, it is reported as an external-damage machine check. (Example: parity error on a channel address or device address transmitted from the I-Unit).

Error Handling Action (General)

The detection of a channel-equipment error causes one of several error-detection latches (EDL) to be set. The setting of these latches and subsequent operations are integrated with channel controls (i.e., channel clocks do not stop). For channel-data checks, no other action is taken after an EDL is set until the normal end of the data transfer in progress. In the case of interface-control checks and channel-control checks, the operation in progress stops immediately.

Error-Detection Latches

The setting of an EDL, plus the channel state at the time of setting, define where the error was detected and what operation was being performed. EDLs are divided into three types, each associated with one of the three channel-equipment error types (CDC, CCC, IFCC).

One set of EDLs serves all channels. When an EDL in a type group has been set, other EDLs of that type are inhibited until a logout occurs. That is, if another error is detected in a channel operation before a logout occurs, that error may not be directly indicated in an EDL. An attempt to set another EDL in a group does cause a "multiple-error" EDL for that group to be set, however. Thus, the only time multiple EDLs can be set within a group is when error conditions are detected simultaneously. The first channel to have its interruption request honored resets all EDLs of all types. The fact that a multiple-error EDL is set in a logout indicates that the logout may be associated with another channel's interruption. On the other hand, a channel can be logged out with no EDL set. One hardware failure may result in many interruptions and, if logout is enabled, in at least one valid logout of EDLs.

Error Logout

After an operation has terminated because of a channel-data or equipment error, the channel makes an interruption request. When the request is honored by the CPU, the C-Unit stores the following information:

- A CSW with status bits set to indicate the error type:

CSW bit 44	CDC
CSW bit 45	CCC
CSW bit 46	IFCC

- A PSW interruption code consisting of channel and unit addresses.

For equipment errors, the channel stores the following additional information:

- I/O Extended Logout (IOEL) – includes error-detection latches, channel state and, for some errors, the contents of the channel buffer storage and subchannel state storage for the channel; cannot be logged unless bit 2 of control register 14 is set (see Figure 9.4); the IOEL field is pointed to by locations 173-175 (Figure 9.8).

Although the clocks continue to run after an IFCC or CCC error, no further interface or storage operations are initiated except as required to report the error. Thus the logged channel state consists of the state at the time of error detection, but possibly modified by posting the result of S-Unit operations initiated before the error was noticed.

Byte Address	Contents		
168	Type	Channel Model No.	Max. IOEL Length
172		IOEL Pointer	
176			
180			
184			I/O Address

Figure 9.8 Fixed Memory Locations

After the I/O extended Logout, a reset occurs. If only the active subchannel was affected by the error, it is reset and a selective-reset interface sequence performed; any interruptions pending in the subchannel are lost. If more than one subchannel was affected, all subchannels on a channel are reset and the system-reset interface sequence is performed; all interruptions pending in the channel are lost.

Errors During I/O Operations

Channel-equipment errors detected during the execution of a START I/O, START I/O FAST RELEASE. TEST I/O, HALT I/O, or HALT DEVICE instruction cause the following actions:

- Two-byte CSW status field (or four-byte CSW with TEST I/O) is stored with CCC or IFCC error indication;
- IOEL is stored if control register 14, bit 2, is set to one;
- Active subchannel or entire channel is reset;
- Condition code is set to one;
- CPU is released.

Errors detected during the execution of a TEST CHANNEL instruction cause an unpredictable condition code setting followed by the procedure described above.

Errors detected during the execution of STORE CHANNEL ID cause the CSW status field to be stored with only the CCC bit equal to one, followed by a condition code setting of one. The condition of main storage locations 168-71 is unpredictable.

When a channel has a channel-equipment error pending, it responds to I/O instructions as follows:

START I/O	CC=2	(Busy)
START I/O FAST RELEASE	CC=2	(Busy)
HALT I/O	CC=0	(Not Working)
HALT DEVICE	CC=0	(Busy)
STORE CHANNEL ID	CC=2	(Busy)
TEST CHANNEL	CC=1	(CSW READY)
TEST I/O	CC=2	(Busy)

In the case of an I/O address or instruction parity error, no logout is performed and condition code 3 is returned (i.e., the parity error looks alike a programming error -- addressing an uninstalled channel).

Input/Output Extended Logout

The I/O extended logout area provides information about the nature of channel-equipment errors and the state of the channel at, or shortly after, the time that the error occurred. Unless masked, it is stored at the time that a CSW with CDC, CCC, or IFCC error indication is stored. If the IOEL mask bit (control register 14, bit 2) is zero, the IOEL is not stored. All other functions associated with the interruption take place, including the resetting of error-detection latches and subchannel or channel controls.

The IOEL area is 48 to 176 bytes long, depending on the number of subchannels installed in a channel. Its origin is pointed to by the address in locations 173-175. The IOEL Pointer can be altered by the program to allow the IOEL area to be relocated. The low-order three bits of the pointer are reserved and ignored by the channel so that the I/O extended logout always begins on a doubleword boundary. The IOEL length is stored in locations 170-171 during execution of a STORE CHANNEL ID instruction (see Figure 9.8).

The IOEL consists of four major areas:

- Error-Detection Latches (EDL). Two words. In conjunction with the channel state, EDLs define when and where the error was detected.
- Channel State Storage (CSS). Two words. This field contains the contents of shifting-state for the channel at the time the error was recognized.
- Channel Buffer Storage (CBS). Eight words. This field contains the control information stored in local channel storage (see Figure 6.1) for the subchannel that was active when the error was recognized.
- Subchannel State Storage (SSS). Zero, 8, 16, or 32 words depending on the number of subchannels. This field contains status information for all subchannels on the channel, packed eight subchannel status digits per word.

For interface-control and for channel-control checks isolatable to a subchannel, the CBS, EDL and CSS fields are stored. For channel-control checks that can't be isolated to a subchannel, all IOEL fields are stored. Channel-data checks cause the EDL, CSS and CBS fields to be stored. The format of these fields is shown in Figure 9.9. Detailed bit descriptions are listed in Figures 9.10-9.15 at the end of this chapter.

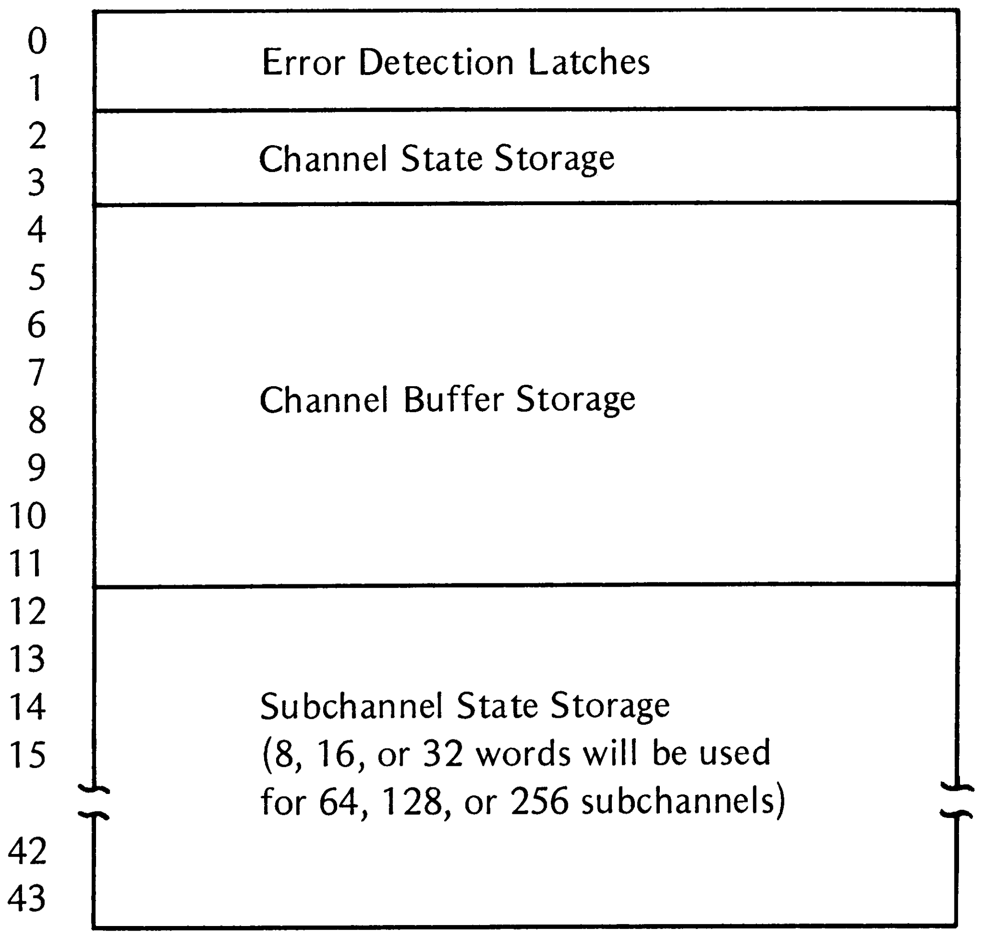


Figure 9.9 IOEL Format

Channel-Data Check

Channel-data check indicates the channel has detected an error in the data transferred between the control unit and main storage during an I/O operation. This information includes read or written data, as well as sense or control data. The error may be detected anywhere inboard the I/O interface. It may be detected between the interface and the channel-data buffers, between the channel-data buffer and the S-Unit interface, or in the S-Unit. Channel-data check is indicated also for errors in data prefetched by the channel, but not actually transmitted to the I/O interface.

A condition indicated as a CDC error causes channel command chaining to be suppressed, but does not stop the current operation. Data transfer is completed normally and an I/O interruption is generated when the device presents channel end. When the CPU honors the interruption request, a CSW is stored with the CDC status bit set. The error-detection latch (including operation control logic information), channel state storage and channel buffer storage fields of the IOEL are also stored if the IOEL mask bit is set to one. The EDL and CSS fields help to isolate the error. Following the storage of a CSW containing an equipment-error check, and after storing an IOEL, if any, the EDLs are reset and the subchannel becomes idle.

CDC Errors in the S-Unit

Channel-data-check errors detected in the S-Unit are reported as soft machine checks. If the error is corrected, the machine-check subclass is system recovery; otherwise, it is external damage. Uncorrectable errors on output data are also reported as channel-data checks. Errors on input data are reported only by the S-Unit.

Parity

When the channel detects a parity error on data during an input operation, good parity is forced on that data sent to the S-Unit. Parity errors detected in the S-Unit on input data cause the storage operation to be aborted; the byte in error is not stored. In addition, adjacent bytes might not be stored.

Channel-Control and Interface-Control Checks

Error Sources

CCC errors are caused by machine malfunctions affecting channel controls. These include parity errors on CCW and data addresses and parity errors on CCW contents. Conditions responsible for a channel control check can cause the contents of the CSW to be invalid and conflicting. The CSW generated by the channel has correct parity, however.

IFCC errors are caused by invalid signals on the I/O interface. These are detected by the channel and usually indicate I/O device malfunctions. They can be caused by the following:

- Invalid parity in the address or status byte received from an I/O device;
- A device responding with an address other than the address specified by the channel during operation initiation;
- A device appearing nonoperable during command chaining;
- A device signal occurring at an invalid time or with invalid duration;
- A device sending a disconnect-in signal to the channel.

Action Taken

An IFCC or CCC condition causes immediate termination of the current operation and generation of an I/O interruption. When the interruption request is honored by the CPU, a CSW is stored with the CCC or IFCC status bit set. An I/O extended logout may also be stored at the time of the interruption if the IOEL mask bit is set to one. The IOEL always includes the error-detection latches, channel state storage, and channel buffer storage fields. Depending on the type of error, the IOEL might also include the subchannel state storage field.

Following storage of the CSW and IOEL, if any, the EDLs are reset. Depending on the type of error, either a selective reset or a channel reset is performed. Selective reset causes the active subchannel to be reset to the idle state and a selective-reset sequence to be performed on the I/O interface. Channel reset causes all subchannels of a channel to be reset to the idle state and the system-reset sequence to be performed on the I/O interface.

DETAILED LOGOUT DESCRIPTIONS

This section contains bit-by-bit descriptions of the fixed-logout area's region code (locations 260-267) and the input/output extended logout (IOEL) area. The IOEL area is pointed to by the address in locations 173-175 (see Figure 9.8).

Byte	Bit	Error Source
260	0	S-Unit Incoming Address Parity Error
	1	S-Unit Tag Key Parity Error
	2	S-Unit Tag ID Error
	3	S-Unit Store Data Parity Error
	4	Main Store Read Address Parity Error
	5	Main Store Key Input Parity Error
	6	Main Store Write Address Parity Error
	7	S-Unit Failing Storage Address Valid
261	0	S-Unit Move Out Data Parity Error Byte 0
	1	S-Unit Move Out Data Parity Error Byte 1
	2	S-Unit Move Out Data Parity Error Byte 2
	3	S-Unit Move Out Data Parity Error Byte 3
	4	S-Unit Primary (1)/Alternate (0) Buffer
	5	S-Unit Channel Request
	6	Main Store Sequence Error
	7	External C-Unit Instruction Code Parity Error
262	0	E-Unit Multiplier Residue Error
	1	E-Unit Adder Residue Error
	2	E-Unit Luck 1 Byte 0 Parity Error
	3	E-Unit Luck 1 Byte 1 Parity Error
	4	E-Unit Luck 1 Byte 2 Parity Error
	5	E-Unit Luck 1 Byte 3 Parity Error
	6	E-Unit Luck 2 Byte 0 Parity Error
	7	E-Unit Luck 2 Byte 1 Parity Error
263	0	E-Unit Luck 2 Byte 2 Parity Error
	1	E-Unit Luck 2 Byte 3 Parity Error
	2	E-Unit Multiplicand Byte 0 Parity Error
	3	E-Unit Multiplicand Byte 1 Parity Error
	4	E-Unit Multiplicand Byte 2 Parity Error
	5	E-Unit Multiplicand Byte 3 Parity Error
	6	E-Unit Adder High-Input Phase Error
	7	E-Unit Adder Low-Input Phase Error
264	0	E-Unit Multiplier Byte Parity Error
	1	E-Unit Byte Adder Input 1 Parity Error
	2	E-Unit Byte Adder Input 2 Parity Error
	3	E-Unit Byte Adder Input 3 Parity Error
	4	External C-Unit Channel and Unit Address Parity Error
	5	External C-Unit Interface Parity Error
	6	C-Unit Channel and Unit Address Parity Error
	7	C-Unit Instruction Code Parity Error
265	0	I-Unit Result Byte 0 Parity Error
	1	I-Unit Result Byte 1 Parity Error
	2	I-Unit Result Byte 2 Parity Error
	3	I-Unit Result Byte 3 Parity Error
	4	I-Unit Control Register and PSW A Parity Error (SD)
	5	I-Unit EAR Parity Error
	6	I-Unit Instruction Stream Entrance Parity Error
	7	I-Unit Instruction Stream Exit Parity Error
266	0-7	Unused
267	0-7	Unused

Figure 9.10 Fixed-Logout Area Region Code

Word	Bit	EDL	Condition
0	0	S-Unit Uncorrectable Data Error	Error detected in S-Unit; valid for both input and output because a channel store into main memory will not validate storage with a previous uncorrectable error.
	1	Store Data	Error detected in S-Unit data registers on data correctly read from local channel storage (LCS).
	2	Fetch Data	On input: error detected on output of LCS during fetch of data from the LCS. On output: error detected on input of LCS after valid fetch of data from the S-Unit.
	3	Interface Data	On input: error on data as received on the interface detected while writing into the LCS. On output: error on data written onto the interface detected at the output of the LCS.

Figure 9.11 IOEL Error-Detection Latches for CDC Error

Word	Bit	EDL	Type	Condition	Reset
0	4	Interface Signal	IFCC	Invalid signal on interface.	Selective
	5	Status or Address Error	IFCC	Address received from control unit not equal to address sent by channel, or PE on status or address received from control unit.	Selective
	6	Disconnect-In	IFCC	Disconnect-In received.	Selective
	7	Count Store PE	CCC	Count PE at input to LCS.	Selective
	8	Count Fetch PE	CCC	Count PE at output of LCS.	Selective
	9	DA Store PE	CCC	Data Address PE at input to LCS.	Selective
	10	DA Fetch PE	CCC	Data Address PE at output of LCS.	Selective
	11	SU Op Error on Data	CCC	S-Unit Signaled Operation Error on Data Transfer.	Selective
	12	Key Error on Data	CCC	S-Unit Signaled key error on data transfer.	Selective
	13	Illegal GTS	CCC	Invalid code in GTS.	System

Figure 9.12 Error-Detection Latches for CCC/IFCC Errors (1 of 2)

Word	Bit	EDL	Type	Condition	Reset
0	14	Operation Control Error	CCC	Error exit called for in Operation Control Logic	Selective
	15	Fetch Data PE	CCC	PE detected in Fetch Data Register.	Selective
	16	S-Unit Data PE	CCC	PE detected in S-Unit Data Register.	Selective
	17	Data Holding Reg. PE	CCC	PE detected in Data Holding Register.	Selective
	18-19	Reserved			
	20	Multiple Data Checks			
	21	Multiple IFCCs			
	22	Multiple CCCs			
	23	Reserved			
	24-31	PTR Procedure Local State			
1	0-26	PTR Procedure Specification			
	27-28	Reserved			
	29-31	PTR Parameter Specification			

Figure 9.12 IOEL Error-Detection Latches for CCC/IFCC Errors (2 of 2)

Word	Bits	Status	
2	0-5	Global transfer state	
		000000	Idle
		001	Operation Pending
		010	Accept Status
		011	Accept Status 1
		100	System Reset
		101	Selective Reset
		110	Reset Sequence 1
		111	Reset Sequence 2
		001000	Polling Sequence 1
		001	Polling Sequence 2
		010	Polling Sequence 3
		011	Polling Sequence 4
		100	Previous State to Status Ready
		101	Restart
		110	Status Ready
		111	Unused
		010000	Interface Disconnect 1
		001	Interface Disconnect 2
		010	Interface Disconnect 3
		011	Interface Disconnect 4
		100	Interface Disconnect 5
		101	Interface Disconnect 6
		110	Disconnected
		111	Lock
		011000	Accept and disconnect
		001	Accept and disconnect 1
		010	Accept and disconnect 2
		011	Accept and disconnect 3
		100	Halted, incorrect length
		101	Previous State to Status Ready, Incorrect length
		110	Status Ready, Incorrect Length
		111	Unused
		100000	Stack
		001	Stack 1
		010	Control Unit Busy 2
		011	Control Unit Busy 1
		100	Control Unit Busy Status Ready
		101	Initial Status Ready
		110	System Reset 1
		111	System Reset 2
		101000	Unused
		001	Unused
		010	Unused
		011	Unused
		100	Select Sequence
		101	Select Sequence 1
		110	Select Sequence 2
		111	Select Sequence 3

Figure 9.13 IOEL Channel State Storage (1 of 3)

Word	Bits	Status
2	0-5	110000 Select Sequence 4
		001 Select Sequence 5
		010 Select Sequence 6
		011 Select Sequence 7
		100 Select Sequence 8
		101 Select Sequence 9
		110 Select Sequence 10
		111 Select Sequence 11
		111000 Buffer Empty
		001 Buffer Empty, Service Out Set 1
		010 Buffer Empty, Service Out Set
		011 Buffer Empty, Service Out Set 2
		100 Buffer Full
		101 Buffer Full, Service Out Set 1
		110 Buffer Full, Service Out Set
		111 Buffer Full, Service Out Set 2
	6-11	OCL Local State
		OCPPPP Wait chain data
		100ETT Initial selection
		101EXX Idle
		1100TT SIOF
		1101TT Contingent hold
		11100X Chain
		11101X Halt
		11110X Transfer
		The bits C, P, E and T are modifier bits for a particular state that describe circumstances of entry. Bits labeled X are not significant.
	12-13	Transfer op
		00 TIO
		01 Output
		10 Input
		11 Input backward
	14-18	Flags
		14 Chain data
		15 Chain command
		16 SLI
		17 Skip; except when OCL Local State is "Wait chain data", then bit 18 is PCI pending
		18 PCI
	19-21	Channel Interrupt State
		000 SSS not scanned
		001 No Interrupts
		010 Global Channel Control Check
		011 Channel Available Interrupt
		100 Secondary Interrupt
		101 Primary Interrupt
		110 PCI
		111 Equipment Error Check
	22	Chaining check
	23	Channel-data check

Figure 9.13 IOEL Channel State Storage (2 of 3)

Word	Bits	Status
2	24-26	Unusual termination field 000 None 001 Halt 010 Program check imminent 011 Protection check imminent 100 Program check 101 Protection check 110 Interface control check 111 Channel control check
	27	Interface transfer width if GTS shows channel connected to device. Retry status flag if GTS shows channel not connected.
	28-29	Channel type 00 Byte multiplexer 01 Block multiplexer 10 Selector 11 Not operational
	30	Interrupt service now
	31	I/O instruction service now
3	0-14	Channel buffer store pointers 0-4 DACL pointer 5-9 Buffer byte count 10-14 Available data buffer pointer
	15-16	Number of subchannels 00 Zero or selector 01 64 subchannels 10 128 subchannels 11 256 subchannels
	17-20	Subchannel number origin
	21	Shared subchannel flag
	22-26	PTR/SU-LCS Control
	27-31	Reserved

Figure 9.13 IOEL Channel State Storage (3 of 3)

Word	Bits	Contents
4	0-7	Command
	8-15	Flags (SCS flags supersede this field)
		8 Chain data
		9 Chain command
		10 SLI
		11 Skip
		12 W bit from SCS; valid on byte multiplexer only
		13 Interface control check detected
		14 Channel data check detected
		15 Chaining check detected
	16-23	Channel number
	24-31	Device address
5	0-3	Storage protection key
	4-7	Zero
	8-31	Data address
6	0-7	Unit status
	8-15	Channel status
	16-31	Count
7	0-3	Storage protection key
	4-5	Zero
	6-7	Deferred condition code
	8-31	Command address
8	0-3	Storage protection key
	4-7	Zero
	8-31	Retry command address
9	0-3	Storage protection key
	4-7	Zero
	8-31	Backup command address
10	0-15	Zero
	16-23	Channel number
	24-31	Interrupt device address buffer
11	0-7	Zero
	8-31	Backup data address/IOEL pointer

Figure 9.14 IOEL Channel Buffer Storage

Word	Bits	Contents
12	0-3	Subchannel 0 state
		1000 Idle
		0001 Disconnected
		0010 Halt I/O
		1101 Primary interrupt pending
		1110 PCI pending, disconnected
		0111 Unit check interrupt pending
	4-7	Subchannel 1 state
	8-11	Subchannel 2 state
		.
		.
		.
43	28-31	Subchannel 255 state

Figure 9.15 IOEL Subchannel State Storage

10. PROGRAMMING CONSIDERATIONS

OPERATING SYSTEMS AVAILABLE

The 470/6 Computing System is supported with a version of the OS/360 operating system. This operating system is distributed to the customer once it has been tailored for the AMDAHL 470/6 System. The only change in code is the modification to the machine-check and channel-check handling routines (recovery management code) normally required when moving to different models within the S/370 line.

Specifically, AMDAHL provides and supports the MFT and MVT control programs of OS/360 and, in addition, supports HASP, for use with these control programs.

The AMDAHL version of this system is upward compatible with S/360 architecture and software. Thus, the AMDAHL 470/6 System user has the great advantage of being able to continue with his present operating system and, consequently, to maintain complete compatibility with his current methods (i.e., programming, data, operations).

COMPATIBILITY

Operating System Compatibility

Because AMDAHL provides the same OS/360 code that IBM provides, operating system compatibility is assured. As described above, the only addition to AMDAHL software occurs in the recovery management component. Here AMDAHL adds code to allow for unique error recovery should the hardware experience a failure. This area is completely the concern of the system.

Since this is the same operating system provided by IBM, all programs running under a given IBM release of OS/360 will also execute properly on the same release from AMDAHL. This, of course, would include modifications to the operating system itself.

Certain S/360 programs rejected by S/370 are also inoperable on AMDAHL equipment. The exceptions, as defined by IBM, are listed in the document *AMDAHL 470 Operating System Support* (G201).

System Console Software

The code required to process the AMDAHL system console's independent functions (such as the system control and maintenance functions described in Chapter 7) is all contained in the console software. OS/360 code is not affected.

OPERATING SYSTEM MAINTENANCE

Software maintenance at AMDAHL provides diagnosis and repair capability for OS/360 (and the commonly-used extensions to this operating system) with minimum interruption to customers.

The commitment to operating system maintenance can be summarized as follows:

- Experienced system engineers responsible for maintaining all AMDAHL-supplied software.
- Prompt temporary repair of all failures in AMDAHL-supplied software.
- A consolidated data base of all maintenance information available from customer sites, program development and independent sources. Comprehensive indexes provide rapid retrieval of information used to identify previously reported problems and the availability of known fixes. This information will be updated and published weekly.

Software maintenance is supported on three levels:

- The system engineer functioning at the customer site;
- The field support center (FSC) located at AMDAHL headquarters;
- The AMDAHL programming staff.

Refer to the document *AMDAHL 470 Operating System Support* (G201) for details.

OPTIMIZING HARDWARE PERFORMANCE

The 32-Byte Block

The AMDAHL 470/6 storage system is organized around a 32-byte block (Chapter 2). Programmers can optimize hardware performance by following these guidelines:

- Iterative loops should be on a block boundary.
- Data accesses by a routine should be compressed into a block.
- Program should ORG on a block boundary (assumes loaded on a line).

- Channel programs should start on a block boundary. (Transfer-in-channel loops should be within a block.)

- Channel data buffers should start on a block boundary.

- Save areas should be on block boundaries.

NOTE: Current OS/360 software is doubleword oriented and these guidelines will be effective only if the Linkage Editor loads on a block boundary.

Instruction Pipeline

In addition, the performance estimates in Chapter 11 and the discussion of the instruction pipeline in Chapter 4 suggest the following guidelines:

- Preload registers to be used in address generation at least three instructions before their usage.

- Registers receiving results in arithmetic processes should not be used for address generation in the two instructions immediately following the arithmetic.

- Branch-on-Condition should be used to branch on the exception rather than the normal condition.

- Branch-on-Condition instructions that are likely to branch should be written to account for the factors listed with the BC instruction timing estimate (Chapter 11).

- SS-type instructions should be used only when necessary.

- Shifting should be used in preference to multiply or divide for powers of two.

- RX instructions execute as rapidly as RR instructions. Less register contention occurs if RX instructions are used.

- Storing in the instruction stream (including the parameter stream) is to be avoided.

- Sequential coding runs faster than iterative or decision-making code because it can be prefetched.

- There is no degradation in using the same register in sequential instructions if it is not modified and used for address generation.

- Add and subtract instructions are faster if they are fixed-point.
- If possible, multiply by a constant rather than divide.
- Halfword instructions execute as rapidly as fullword instructions and the operands require less storage space.
- Fixed-point arithmetic is much faster than decimal and faster than floating-point arithmetic. Single-precision floating-point arithmetic is two to three times faster than double precision.

11. INSTRUCTION PERFORMANCE

GENERAL CONSIDERATIONS

The following list estimates the performance of each instruction in terms of the number of CPU cycles required to perform the indicated operation. The performance estimates of certain instructions contain variables whose effect on instruction timing has not been determined at present. These instructions usually involve the channel unit or the storage-control unit. Where the instructions occur in the following list, the variables are included as part of the performance estimate, but no specific values are given. Explanations of the symbols representing these variables are provided in the legend following the "Performance Estimates" section of this chapter.

The performance figures assume that all addresses requested by fetches and stores are currently valid locations in the high-speed buffer. If any request requires data that are not contained in buffer storage, additional time must be added to the performance value shown here (the time required to fetch data from main storage into the high-speed buffer—see Chapter 2).

PERFORMANCE ESTIMATES

Instruction	*Format*	*Mnemonic*	*Performance (CPU Cycles)*
Add	RR	AR	2
Add	RX	A	2
Add Decimal	SS	AP	13-47* + S1/S2

	Length in Bytes of the Longer Operand			
	1-4	5-8	9-12	13-16
*Signs Alike	13	23	33	43
Signs Different	15	25	36	47

Add Halfword	RR	AH	2
Add Logical	RR	ALR	2
Add Logical	RX	AL	2

Instruction	*Format*	*Mnemonic*	*Performance*
Add Normalized (Extended)	RR	AXR	30-39*

*Basic + Alignment + Recomplement + Post-normalization	
Basic	15
Alignment	
Exponent Difference $<$ 8	9
8 $\leqslant$ Exponent Difference $<$ 16	8
16 $\leqslant$ Exponent Difference	7
Recomplement (if required)	3
Post-normalization	
Leading zero digits $\leqslant$ 6	6
6 $<$ Leading zero digits $\leqslant$ 14	10
14 $<$ Leading zero digits	7

Add Normalized (Long)	RR	ADR	7-9*
Add Normalized (Long)	RX	AD	7-9*

*9 cycles if recomplementation is required; otherwise, 7 cycles.

Add Normalized (Short)	RR	AER	6-7*
Add Normalized (Short)	RX	AE	6-7*

*7 cycles if recomplementation is required; otherwise, 6 cycles.

Add Unnormalized (Long)	RR	AWR	7-9*
Add Unnormalized (Long)	RX	AW	7-9*

*9 cycles if recomplementation is required; otherwise, 7 cycles.

Add Unnormalized (Short)	RR	AUR	6-7*
Add Unnormalized (Short)	RX	AU	6-7*

*7 cycles if recomplementation is required; otherwise, 6 cycles.

AND	RR	NR	2
AND	RX	N	2
AND	SI	NI	4 + S1
AND	SS	NC	6L + S1
Branch and Link	RR	BALR	10
Branch and Link	RX	BAL	10
Branch on Condition	RR	BCR	2-6* + S3
Branch on Condition	RX	BC	2-6* + S3

Instruction	*Format*	*Mnemonic*	*Performance*
*Branch Taken			
Floating-point, early condition code setter			4
Fixed-point, logical early CC setter			5
Late CC setter			6
Branch Not Taken			
Early CC setter (fixed and floating)			2 + S3
Late CC setter			3 + S3
Branch on Count	RR	BCTR	4-7* + S3
Branch on Count	RX	BCT	4-7* + S3
*Branch Taken			7
Branch Not Taken			4 + S3
Branch on Index High	RS	BXH	6-9* + S3
Branch on Index Low or Equal	RS	BXLE	6-9* + S3
*Branch Taken			9
Branch Not Taken			6 + S3
Compare	RR	CR	2
Compare	RX	C	2
Compare Decimal	SS	CP	13-47*

	Length in Bytes of the Longer Operand			
	1-4	5-8	9-12	13-16
*Signs Alike	15	25	36	47
Signs Different	13	23	33	43

Instruction	*Format*	*Mnemonic*	*Performance*
Compare Halfword	RX	CH	2
Compare Logical	RR	CLR	2
Compare Logical	RX	CL	2
Compare Logical	SI	CLI	2
Compare Logical	SS	CLC	4W* + 4X + 4Y

*W equals the number of words compared. If instruction specifies 4 words of length and compare is found in second word, W=2.

Instruction	*Format*	*Mnemonic*	*Performance*
Compare Logical Characters under Mask	RS	CLM	2
Compare Logical Long	RR	CLCL	32 + 4W + 4X + 4Y
Compare (Long)	RR	CDR	3-6*
Compare (Long)	RX	CD	3-6*

Instruction	*Format*	*Mnemonic*	*Performance*

*3 cycles if both operands are normalized; otherwise, 6 cycles.

Compare (Short)	RR	CER	3-4*
Compare (Short)	RX	CE	3-4*

*3 cycles if both operands are normalized; otherwise, 4 cycles.

Convert to Binary	RX	CVB	5-6* + 2D

*5 if sign is positive.
6 if sign is negative.

Convert to Decimal	RX	CVD	42-43* + S2

*42 cycles if operand 1 is positive; 43 cycles if it is negative.

Divide	RR	DR	51
Divide	RX	D	51
Divide Decimal	SS	DP	28-1266* + S1/S2

***If L2 $\leqslant$ 3, formula is 20 + (5 + 4Q)N + 2RW**
If 3 $<$ L2, formula is 27 + (9 + 6Q)N + 2RW
where Q equals the average value of the quotient digits plus 1.
N equals the number of quotient digits.
RW equals the number of words of result.

Divide (Long)	RR	DDR	64-69*
Divide (Long)	RX	DD	64-69*

*Basic + Pre-norm 1 + Pre-norm 2 + Divident Preshift

Basic	64
Pre-normalization of operand 1 (if required)	2
Pre-normalization of operand 2 (if required)	2
Dividend preshift (if required)	1

Divide (Short)	RR	DER	27
Divide (Short)	RX	DE	27
Edit	SS	ED	6L + S1
Edit and Mark	SS	EDMK	6L + S1
Exclusive OR	RR	XR	2
Exclusive OR	RX	X	2
Exclusive OR	SI	XI	4 + S1
Exclusive OR	SS	XC	6L + S1
Execute	RX	EX	20 + TGE
Halt I/O	SI	HIO	10 + CURT

Instruction	*Format*	*Mnemonic*	*Performance*
Halt Device	SI	HDV	10 + CURT
Halve (Long)	RR	HDR	6
Halve (Short)	RR	HER	4
Insert Character	RX	IC	2
Insert Character under Mask	RS	ICM	9
Insert Storage Key	RR	ISK	2 + MS
Load	RR	LR	2
Load	RX	L	2
Load Address	RX	LA	2
Load and Test	RR	LTR	2
Load and Test (Long)	RR	LTDR	2
Load and Test (Short)	RR	LTER	2
Load Complement	RR	LCR	2
Load Complement (Long)	RR	LCDR	2
Load Complement (Short)	RR	LCER	2
Load Control	RS	LCTL	8 + 2R + CURT
Load Halfword	RX	LH	2
Load (Long)	RR	LDR	2
Load (Long)	RX	LD	2 + S3
Load Multiple	RS	LM	2R
Load Negative	RR	LNR	2
Load Negative (Long)	RR	LNDR	2
Load Negative (Short)	RR	LNER	2
Load Positive	RR	LPR	2
Load Positive (Long)	RR	LPDR	2
Load Positive (Short)	RR	LPER	2
Load PSW	SI	LPSW	14 + CURT + 64WST
Load Rounded (Extended to Long)	RR	LRDR	5-6*

*6 cycles if rounding is required; otherwise, 5 cycles.

Load Rounded (Long to Short)	RR	LRER	4
Load (Short)	RR	LER	2
Load (Short)	RX	LE	2
Move	SI	MVI	2 + S1
Move	SS	MVC	6 + MV* + S1

*MV equals

No overlap or overlap > 32 bytes or both operands on word boundary	4W
3 bytes < overlap ≤ 32 bytes or both operands not on word boundary	5W
1 byte < overlap ≤ 3 bytes	4B
overlap = 1 byte	6B

Instruction	*Format*	*Mnemonic*	*Performance*
Move Long	RR	MVCL	32 + 4W
Move Numerics	SS	MVN	6B + S1
Move with Offset	SS	MVO	2 + MV* + S1

*MV equals:	
No overlap or overlap ≠ 1 byte	4B
Overlap = 1 byte	6B

Move Zones	SS	MVZ	6B + S1
Multiply	RR	MR	7
Multiply	RX	M	7
Multiply Decimal	SS	MP	21-409* + S1/S2

*This cycle estimation assumes a random distribution of multiplier digits. If the multiplier data is not random, the performance can vary greatly. The number of cycles required increases as the multiplier digits go from a value of zero to a value of four, or go from a value of nine to a value of five. The cycles shown in the table below represent an average.

		Multiplier Length in Bytes								
		1	2	3	4	5	6	7	8	
Multiplicand Length in Bytes	1-4	19	37	95	–	–	–	–	–	+2
	5-8	27	57	87	117	140	160	180	–	+4
	9-12	38	82	126	170	205	235	265	295	+4
	13-16	30	108	166	224	270	314	358	402	+6
		+0				+1				
To get new values add the indicated row and column values.										

Multiply (Extended	RR	MXR	94-98*

*Basic + Pre-norm 1 + Pre-norm 2	
Basic	94
If 6< leading zero digits in OP1 ⩽ 14	2
Else	0
If 6< leading zero digits in OP2 ⩽ 14	2
Else	0

Multiply Halfword	RX	MH	2
Multiply (Long)	RR	MDR	20-24*
Multiply (Long)	RX	MD	20-24*

Instruction	*Format*	*Mnemonic*	*Performance*

*Basic + Pre-norm 1 + Pre-norm 2	
Basic	20
Pre-normalization of Operand 1	2
Pre-normalization of Operand 2	2

Multiply (Long to Extended)	RR	MXDR	28-31*
Multiply (Long to Extended)	RX	MXD	28-31*

*Basic + Pre-norm 1 + Pre-norm 2	
Basic	28
Pre-normalization of Operand 1	2
Pre-normalization of Operand 2	1

Multiply (Short)	RR	MER	8
Multiply (Short)	RX	ME	8
OR	RR	OR	2
OR	RX	O	2
OR	SI	OI	4 + S1
OR	SS	OC	6L + S1
Pack	SS	PACK	4B + 2PK* + S1

*PK equals 1-16 if overlap is present; it is a function of the number of interline overlaps.

Read Direct	SI	RDD	38 + HRT
Set Clock	SI	SCK	6 + TODRT
Set Program Mask	RR	SPM	5
Set Storage Key	RR	SSK	15 + BR
Set System Mask	SI	SSM	10 + CURT
Shift and Round Decimal	SS	SRP	40 + S2
Shift Left Double	RS	SLDA	4
Shift Left Double-Logical	RS	SLDL	3
Shift Left Single	RS	SLA	2
Shift Left Single-Logical	RS	SLL	2
Shift Right Double	RS	SRDA	3
Shift Right Double-Logical	RS	SRDL	2
Shift Right Single	RS	SRA	2
Shift Right Single-Logical	RS	SRL	2
Start I/O	SI	SIO	10 + CURT
Start I/O Fast Release	SI	SIOF	10 + CURT
Store	RX	ST	2 + S1
Store Channel ID	SI	STIDC	23 + CURT
Store Character	RX	STC	2 + S1
Store Character under Mask	RS	STCM	4 + S1
Store Clock	SI	STCK	6 + S2 + TODRT
Store Control	RS	STCTL	2R + S2
Store CPU ID	SI	STIDP	4 + S2

Instruction	*Format*	*Mnemonic*	*Performance*
Store Halfword	RX	STH	2 + S1
Store (Long)	RX	STD	4 + S2
Store Multiple	RS	STM	2R + S2
Store (Short)	RX	STE	2 + S1
Subtract	RR	SR	2
Subtract	RX	S	2
Subtract Decimal	SS	SP	13-47* + S1/S2

	Length in Bytes of the Longer Operand			
	1-4	5-8	9-12	13-16
*Signs Alike	15	25	37	47
Signs Different	13	23	33	43

Subtract Halfword	RX	SH	2
Subtract Logical	RR	SLR	2
Subtract Logical	RX	SL	2
Subtract Normalized (Extended)	RR	SXR	30-39*

*Basic + Alignment + Recomplement + Post-normalization	
Basic	15
Alignment	
Exponent Difference < 8	9
8 ≤ Exponent Difference <16	8
16 ≤ Exponent Difference	7
Recomplement (if required)	3
Post-normalization	
Leading zero digits < 6	6
6 ≤ Leading zero digits ≤ 14	10
14 < Leading zero digits	7

Subtract Normalized (Long)	RR	SDR	7-9*
Subtract Normalized (Long)	RX	SD	7-9*

*9 cycles if recomplementation is required; otherwise, 7 cycles.

Subtract Normalized (Short)	RR	SER	6-7*
Subtract Normalized (Short)	RX	SE	6-7*

*7 cycles if recomplementation is required; otherwise, 6 cycles.

Subtract Unnormalized (Long)	RR	SWR	7-9*
Subtract Unnormalized (Long)	RX	SW	7-9*

*9 cycles if recomplementation is required; otherwise, 7 cycles.

Instruction	*Format*	*Mnemonic*	*Performance*
Subtract Unnormalized (Short)	RR	SUR	6-7*
Subtract Unnormalized (Short)	RX	SU	6-7*

*7 cycles if recomplementation is required; otherwise, 6 cycles.

Instruction	*Format*	*Mnemonic*	*Performance*
Supervisor Call	RR	SVC	30 + CURT
Test and Set	SI	TS	2 + SUI
Test Channel	SI	TCH	10 + CURT
Test I/O	SI	TIO	10 + CURT
Test Under Mask	SI	TM	2
Translate	SS	TR	$5\lceil B/4\rceil$* + 2 + 4B + S1
Translate and Test	SS	TRT	$5\lceil B/4\rceil$* + 4B + 4M* + 4

*M equals 1 if any other than last byte to be tested is nonzero;
otherwise, M=0
Term within $\lceil B/4\rceil$ is greatest integer; e.g., if B=1, term=1.

Instruction	*Format*	*Mnemonic*	*Performance*
Unpack	SS	UNPK	$5\lceil (B-1)/2\rceil$* + 4 + 2UPK* + S1

*UPK equals 1 if overlap occurs.
Term within $\lceil (B-1)/2\rceil$ is greatest integer.

Instruction	*Format*	*Mnemonic*	*Performance*
Write Direct	SI	WRD	36
Zero and Add	SS	ZAP	13 + 5WL

LEGEND

Symbol	Explanation
B	Number of bytes moved, packed/unpacked, or translated.
BR	Storage-control unit buffer release time.
CURT	Channel unit release time.
D	Number of significant decimal digits.
HRT	Hold response time.
L	In SS-type instructions the L field specifies the length in bytes of the first operand.
L2	In SS-type instructions the L2 field specifies the length in bytes of the second operand.
MS	Time required to move a line from main storage.
R	Number of registers to be loaded or stored.
SUI	Storage-control unit interlock time.
S1	S1 = 2 if M = 6 = 1 if M = 7 = 0 if M = 8 where M equals the number of execution cycles attributable to the three words of the instruction stream following the instruction of interest.
S2 S3	S2 = 0 if P = 8 = 1 if P = 7 = 2 if P = 6 = 3 if P = 5 = 4 if P = 4 S3 = 0 if P = 6 = 1 if P = 5 = 2 if P = 4 where P equals the number of execution cycles attributable to the two words of the instruction stream following the instruction of interest.
S1/S2	Use the S1 calculation if both operands total less than five bytes; otherwise, use S2.

Symbol	Explanation
TGE	Target instruction execution time.
TODRT	Time-of-day response time.
W	Number of words compared or moved.
WL	Number of words in the longer operand.
WST	Include this calculation if the wait-state bit in the new PSW is set.
X	In logical comparison (CLC, CLCL), X = 1 if an unequal is found in other than the last word compared; otherwise, X = 0.
Y	In logical comparison (CLC, CLCL), Y = 1 if an unequal is found in other than the last two words compared; otherwise, Y = 0.

EARLY/LATE CONDITION CODE SETTINGS

The timing estimate for the Branch-on-Condition instruction depends partially on how early in its execution the tested instruction sets the condition code. In the following table, "E" represents an instruction that sets the condition code "early" and "L" a "late" condition-code setter. "ENI,L" means the instruction will set the condition code "early" if its input operands are normalized; otherwise, the setting is delayed to "late." An early condition code setting will reduce instruction execution time by 1-2 cycles.

Instruction	*Mnemonic*	*CC Setting*
Add	AR	E
Add	A	E
Add Decimal	AP	E
Add Halfword	AH	E
Add Logical	ALR	E
Add Logical	AL	E
Add Normalized (Extended)	AXR	E
Add Normalized (Long)	ADR	ENI,L
Add Normalized (Long)	AD	ENI,L
Add Normalized (Short)	AER	ENI,L
Add Normalized (Short)	AE	ENI,L
Add Unnormalized (Long)	AWR	ENI,L
Add Unnormalized (Long)	AW	ENI,L
Add Unnormalized (Short)	AUR	ENI,L
Add Unnormalized (Short)	AU	ENI,L
AND	NR	E
AND	N	E
AND	NI	E
AND	NC	E
Compare	CR	E
Compare	C	E
Compare Decimal	CP	E
Compare Halfword	CH	E
Compare Logical	CLR	E
Compare Logical	CL	E
Compare Logical	CLI	E
Compare Logical	CLC	E

Instruction	*Mnemonic*	*CC Setting*
Compare Logical Characters Under Mask	CLM	L
Compare Logical Long	CLCL	E
Compare (Long)	CDR	E
Compare (Long)	CD	E
Compare (Short)	CER	E
Compare (Short)	CE	E
Edit	ED	E
Edit and Mask	EDMK	E
Exclusive OR	XR	E
Exclusive OR	X	E
Exclusive OR	XI	E
Exclusive OR	XC	E
Halt I/O	HIO	E
Insert Character Under Mask	ICM	E
Load and Test	LTR	E
Load and Test (Long)	LTDR	L
Load and Test (Short)	LTER	E
Load Complement	LCR	E
Load Complement (Long)	LCDR	L
Load Complement (Short)	LCER	E
Load Negative	LNR	E
Load Negative (Long)	LNDR	L
Load Negative (Short)	LNER	E
Load Positive	LPR	E
Load Positive (Long)	LPDR	L
Load Positive (Short)	LPER	E
Move Long	MVCL	E
OR	OR	E
OR	O	E
OR	OI	E
OR	OC	E
Shift Left Double	SLDA	L
Shift Left Single	SLA	L
Shift Right Double	SRDA	L
Shift Right Single	SRA	L
Start I/O	SIO	E
Subtract	SR	E
Subtract	S	E
Subtract Decimal	SP	E
Subtract Halfword	SH	E
Subtract Logical	SLR	E
Subtract Logical	SL	E
Subtract Normalized (Extended)	SXR	E

Instruction	*Mnemonic*	*CC Setting*
Subtract Normalized (Long)	SDR	ENI,L
Subtract Normalized (Long)	SD	ENI,L
Subtract Normalized (Short)	SER	ENI,L
Subtract Normalized (Short)	SE	ENI,L
Subtract Unnormalized (Long)	SWR	ENI,L
Subtract Unnormalized (Long)	SW	ENI,L
Subtract Unnormalized (Short)	SUR	ENI,L
Subtract Unnormalized (Short)	SU	ENI,L
Test and Set	TS	E
Test Channel	TCH	E
Test I/O	TIO	E
Test Under Mask	TM	E
Translate and Test	TRT	E
Zero and Add	ZAP	E

NONRETRYABLE INSTRUCTIONS

CPU recovery action from machine-detected instruction errors is described in Chapter 9. This section lists those instructions that are completely nonretryable, or that become nonretryable after a certain threshold has been passed (i.e., after they have modified the machine state).

For example, the BRANCH AND LINK (BAL) instruction has a threshold of two cycles. If the error is detected during the first two cycles of BAL processing, the instruction is retryable; after two cycles it is nonretryable. COMPARE LOGICAL LONG (CLCL) has a threshold of END-8 cycles. This means the instruction cannot be retried if an error is detected in the last eight cycles of CLCL processing. For decimal arithmetic instructions **like ADD DECIMAL (AP), the "END" factor is two cycles for each result word to be** stored (END-2/Result Word). Finally, instructions that store in the instruction stream are not retryable (Threshold=0).

Instruction	*Mnemonic*	*Threshold (Cycles)*
Add Decimal	AP	END-2/Result Word
Add Normalized (Extended)	AXR	2
AND	NI	2
AND	NC	4
Branch and Link	BALR	2
Branch and Link	BAL	2
Branch on Condition	BCR (Taken)	0
Branch on Condition	BC (Taken)	0
Branch on Count	BCTR	2
Branch on Count	BCT	2
Branch on Index High	BXH	2
Branch on Index Low or Equal	BXLE	2
Compare Logical Long	CLCL	END-8
Convert to Decimal	CVD	Store in Instruction Stream
Divide Decimal	DP	END- /Result Word
Edit	ED	4
Edit and Mark	EDMK	4
Exclusive OR	XI	2
Exclusive OR	XC	4
Halt I/O	HIO	2
Load Control	LCTL	2
Load Multiple	LM	2
Load PSW	LPSW	2
Move	MVI	Store in Instruction Stream
Move	MVC	6
Move Long	MVCL	END-8
Move Numerics	MVN	4

Instruction	*Mnemonic*	*Threshold (Cycles)*
Move with Offset	MVO	2
Move Zones	MVZ	4
Multiply Decimal	MP	END-2/Result Word
Multiply (Extended)	MXR	2
Multiply (Long to Extended)	MXDR	2
Multiply (Long to Extended)	MXD	2
OR	OI	2
OR	OC	4
Pack	PACK	2
Read Direct	RDD	2
Set Storage Key	SSK	Store in Instruction Stream
Set System Mask	SSM	2
Shift and Round Decimal	SRP	Store in Instruction Stream
Start I/O	SIO	2
Start I/O Fast Release	SIOF	2
Store	ST	Store in Instruction Stream
Store Channel ID	STIDC	2
Store Character	STC	Store in Instruction Stream
Store Character under Mask	STCM	Store in Instruction Stream
Store Clock	STCK	Store in Instruction Stream
Store Control	STCTL	Store in Instruction Stream
Store CPU ID	STIDP	Store in Instruction Stream
Store Halfword	STH	Score in Instruction Stream
Store (Long)	STD	Store in Instruction Stream
Store Multiple	STM	Store in Instruction Stream
Store (Short)	STE	Store in Instruction Stream
Subtract Decimal	SP	END-2/Result Word
Subtract Normalized (Extended)	SXR	2
Test and Set	TS	0
Test Channel	TCH	2
Test I/O	TIO	2
Translate	TR	2
Translate and Test	TRT	2
Unpack	UNPK	2
Write Direct	WRD	2
Zero and Add	ZAP	END-2/Result Word

APPENDIX A: ABBREVIATIONS

APL	Automatic Program Load
ASCII	American Standard Code for Information Interchange
B	Backup Bit
CAW	Channel Address Word
CBS	Channel Buffer Storage
CC	Condition Code
CC.	Command Chain
CCC	Channel-Control Check
CCI	Computer-to-Console Interface
CCW	Channel Command Word
CD	TOD-Clock-Damage Indicator Bit
CD	Data Chain
CDC	Channel-Data Check
CPU	Central Processing Unit
CR	Control Registers Validity Bit
CRT	Cathode-Ray Tube Display Console
CSS	Channel State Storage
CSW	Channel Status Word
C-Unit	Channel Unit
D	Delay Bit
DA	Data Address
DACL	Direct Access Control Logic
EAR	Effective Address Generation Register
EBCDIC	Extended Binary-Coded-Decimal Interchange Code
EC	Extended Control (Mode)
ECC	Error Checking and Correction
ED	External-Damage Indicator Bit
EDL	Error-Detection Latch
EM	External-Damage Report Mask Bit
E-Unit	Execution Unit
FA	Failing-Storage-Address Validity Bit
FE	AMDAHL Field Engineer
FP	Floating-Point Registers Validity Bit
FPR	Floating-Point Register
FSA	Failing Storage Address
GPR	General-Purpose Register
GR	General-Purpose Registers Validity Bit

GTS	Global Transfer State
HS	Hard-Stop Mask Bit
HSB	High-Speed Buffer
IA	PSW Instruction-Address Validity Bit
IAR	Instruction Counter
IBR	Instruction Buffer Register
IFCC	Interface-Control Check
IM	Input/Output Extended Logout Mask Bit
I/O	Input/Output
IOEL	Input/Output Extended Logout
IPL	Initial Program Loader
I-Unit	Instruction Unit
IWR	Instruction Word Register
KB	Kilobyte (1000 bytes)
KE	Key-In-Storage Error-Uncorrected Bit
LCS	Local Channel Storage
LSI	Large-Scale Integrated Circuit
LUCK	E-Unit Logical Component
MB	Megabyte (1,000,000 bytes)
MCI	Machine-Check Interruption
MCIC	Machine-Check Interruption Code
MFT	Multiprogramming with a Fixed Number of Tasks
MS	PSW Masks and Key Validity Bits
MSAR	Main-Storage Address Register
MTB	Memory Transfer Bus
MVT	Multiprogramming with a Variable Number of Tasks
NS	Nanosecond
OCL	Operations Control Logic
OPSR	Operating-State Register
PCI	Program Control Interrupt
PD	Instruction-Processing-Damage Indicator Bit
PDU	Power Distribution Unit
PE	Parity Error
PM	PSW Program Mask and CC Validity Bit
POP	Power-On Initialization Program
PRT	Procedure Transfer Register
PSW	Program Status Word
RM	Recovery-Report Mask Bit
RMS	Recovery Management Support
SBS	Subchannel Buffer Storage
SC	Storage-Error-Corrected Indicator Bit

SCS	Shifting Channel State
SD	System-Damage Indicator Bit
SE	Storage-Error-Uncorrected Indicator Bit
SLI	Suppress Length Indicator
SR	System-Recovery Indicator Bit
SSM	Set System Mask
SSS	Subchannel State Storage
S-Unit	Storage-Control Unit
TD	Timer-Damage Indicator Bit
TOD	Time-of-Day Clock
WP	PSW AMWP (Bits 12-15) Validity Bit

INDEX

No. G009-1

READER'S COMMENTS

AMDAHL 470/6
Machine Reference Manual

Name (Optional) ______________________________

Address ______________________________

Job Title ______________________________

Did you find this publication:	GOOD	FAIR	POOR
Well organized?	____	____	____
Easily read?	____	____	____
Well illustrated?	____	____	____
Accurate?	____	____	____
Complete?	____	____	____
Adequate to meet your needs?	____	____	____

What additional information do you need? ______________________________

____

____

____

How can we improve this document? ______________________________

____

____

____

I need more information on the AMDAHL 470 System. Please contact me. ☐

COMMENTS PLEASE . . .

This publication is one of a series describing the AMDAHL 470 System. Your comments on the reverse side will help us produce better publications.

If you require more information on the AMDAHL 470 System please check the appropriate box and we will contact you.

fold　　　　　　　　　　　　　　　　　　　　fold

First Class
Permit No. 741
Sunnyvale, CA

BUSINESS REPLY MAIL

No Postage Stamp Necessary if Mailed in U.S.A.

Postage will be paid by:

AMDAHL Corporation

1180 Kern Avenue

Sunnyvale, CA 94086

Attention: AMDAHL Publications
Department 353

fold　　　　　　　　　　　　　　　　　　　　fold

amdahl Corporation, Sunnyvale, California