60420000

CONTROL DATA CORPORATION

CONTROL DATA® CYBER 170 COMPUTER SYSTEMS

HARDWARE REFERENCE MANUAL

CENTRAL PROCESSOR INSTRUCTION INDEX

Code	Page	Code	Page	Code	Page	Code	Page
00	4-4	20	4-21	40	4-35	60	4-48
01	4-5	21	4-21	41	4-36	61	4-48
02	4-10	22	4-22	42	4-37	62	4-49
03	4-10	23	4-23	43	4-38	63	4-49
04	4-15	24	4-25	44	4-38	64	4-49
05	4-15	25	4-26	45	4-39	65	4-49
06	4-16	26	4-27	46	4-40	66	4-50
07	4-16	27	4-28	47	4-44	67	4-50
10	4-17	30	4-29	50	4-44	70	4-50
11	4-17	31	4-30	51	4-45	71	4-51
12	4-18	32	4-31	52	4-45	72	4-51
13	4-18	33	4-31	53	4-46	73	4-51
14	4-19	34	4-32	54	4-46	74	4-52
15	4-19	35	4-33	55	4-47	75	4-52
16	4-20	36	4-34	56	4-47	76	4-52
17	4-20	37	4-34	57	4-48	77	4-52

PERIPHERAL PROCESSOR INSTRUCTION INDEX

Code	Page	Code	Page	Code	Page	Code	Page
00	4-57	20	4-61	40	4-66	60	4-71
01	4-57	21	4-62	41	4-66	61	4-71
02	4-57	22	4-62	42	4-67	62	4-72
03	4-58	23	4-62	43	4-67	63	4-72
04	4-58	24	4-63	44	4-67	64	4-73
05	4-58	25	4-63	45	4-67	65	4-73
06	4-59	26	4-63	46	4-68	66	4-73
07	4-59	27	4-64	47	4-68	67	4-74
10	4-59	30	4-64	50	4-68	70	4-74
11	4-60	31	4-64	51	4-69	71	4-75
12	4-60	32	4-65	52	4-69	72	4-75
13	4-60	33	4-65	53	4-69	73	4-76
14	4-60	34	4-65	54	4-69	74	4-76
15	4-61	35	4-65	55	4-70	75	4-77
16	4-61	36	4-66	56	4-70	76	4-77
17	4-61	37	4-66	57	4-70	77	4-78

REVISION RECORD

REVISION	DESCRIPTION
01	Preliminary release
(12-73)	
02	Major revision; this edition obsoletes previous edition.
(3-74)	
03	Manual revised; includes manual corrections and engineering changes. Pages affected are:
(8-74)	v, 1-1, 2-1, 2-2, 2-5, 2-6, 2-13, 2-15, 2-17, 2-26 through 2-32, 3-1, 4-4, 4-5, 4-6, 4-9
	through 4-15, 4-39, 5-2, 5-6, 5-18, 5-29 through 5-39, 5-42, 5-44, 5-47, A-1 through A-5,
	A-7. Pages added: 2-28.1, 2-33, 2-34, 2-35.
A	Manual released; this edition obsoletes all previous editions and publication no. 19981200.
(5-9-75)	
B	Manual revised; includes Engineering Change Order 36538. Pages 5-4, 5-5, and A-1 through
(7-7-75)	A-10 are revised. Page A-11 is added.
C	Manual revised; includes Field Change Order PD1346. Cover and pages 2-57 and 5-40 are revised.
(9-1-75)	
Publication No. 60420000	

Address comments concerning this manual to:

Control Data Corporation
Publications and Graphics Division
4201 Lexington Avenue North
St. Paul, Minnesota 55112

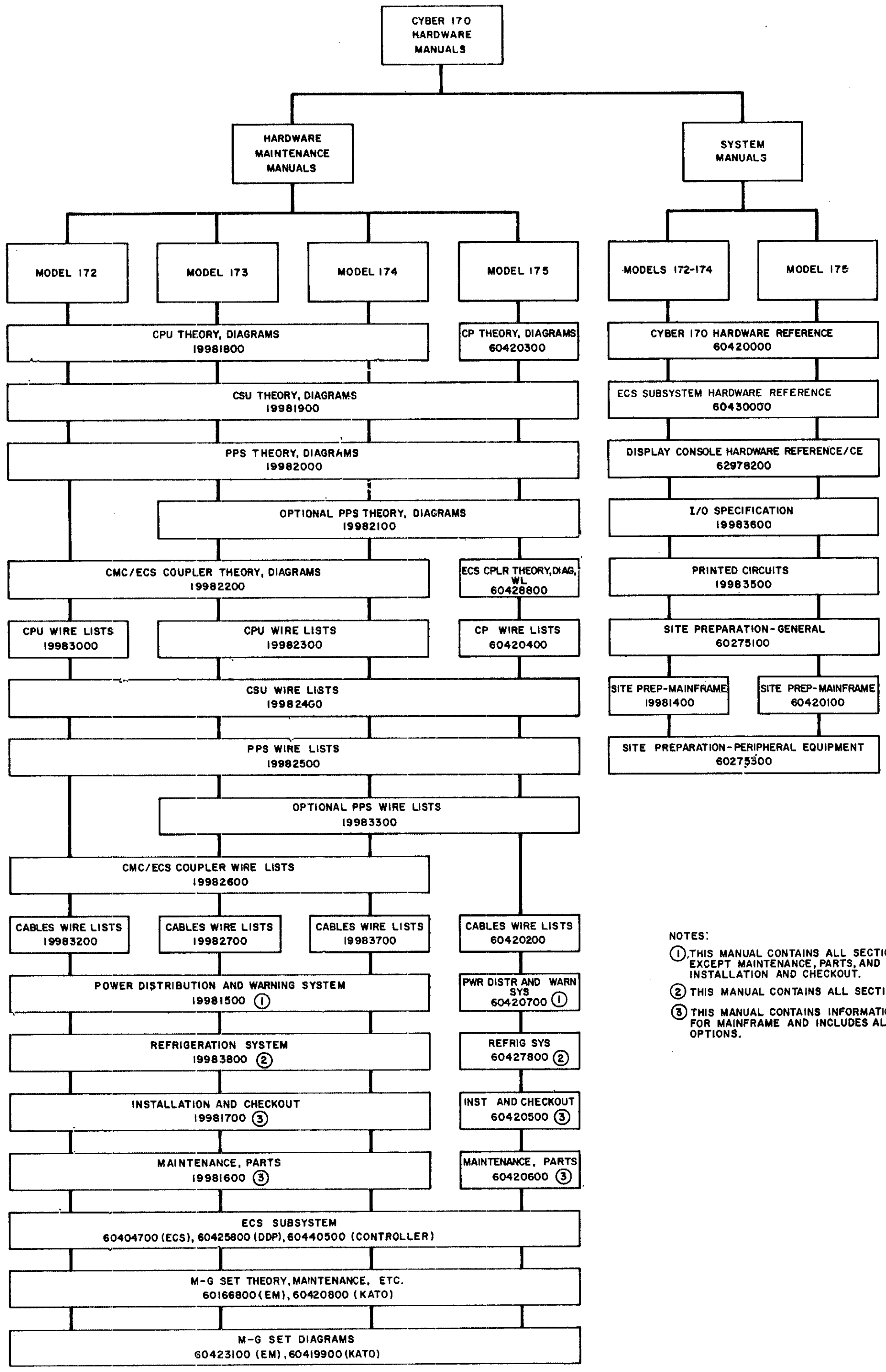
or use Comment Sheet in the back of this manual.

© 1973, 1974, 1975
by Control Data Corporation
Printed in the United States of America

PREFACE

This manual contains hardware reference information for the CONTROL DATA® CYBER 170 Computer Systems, Models 172 through 175. The information describes the systems, procedures for their operation, instructions, and programming information. Additional system hardware information is available in publications listed in the following system publication index.

Publication ordering information and latest revision levels are available from the Literature Distribution Services catalog, publication no. 90310500.


CYBER 170 HARDWARE MANUALS
HARDWARE MAINTENANCE MANUALS
SYSTEM MANUALS
MODEL 172
MODEL 173
MODEL 174
MODEL 175
MODELS 172-174
MODEL 175
CPU THEORY, DIAGRAMS 19981800
CP THEORY, DIAGRAMS 60420300
CYBER 170 HARDWARE REFERENCE 60420000
CSU THEORY, DIAGRAMS 19981900
ECS SUBSYSTEM HARDWARE REFERENCE 60430000
PPS THEORY, DIAGRAMS 19982000
DISPLAY CONSOLE HARDWARE REFERENCE/CE 62978200
OPTIONAL PPS THEORY, DIAGRAMS 19982100
I/O SPECIFICATION 19983600
CMC/ECS COUPLER THEORY, DIAGRAMS 19982200
ECS CPLR THEORY, DIAG, WL 60428800
PRINTED CIRCUITS 19983500
CPU WIRE LISTS 19983000
CPU WIRE LISTS 19982300
CP WIRE LISTS 60420400
SITE PREPARATION-GENERAL 60275100
CSU WIRE LISTS 19982400
SITE PREP-MAINFRAME 19981400
SITE PREP-MAINFRAME 60420100
PPS WIRE LISTS 19982500
SITE PREPARATION-PERIPHERAL EQUIPMENT 60275300
OPTIONAL PPS WIRE LISTS 19983300
CMC/ECS COUPLER WIRE LISTS 19982600
CABLES WIRE LISTS 19983200
CABLES WIRE LISTS 19982700
CABLES WIRE LISTS 19983700
CABLES WIRE LISTS 60420200
POWER DISTRIBUTION AND WARNING SYSTEM 19981500 ①
PWR DISTR AND WARN SYS 60420700 ①
REFRIGERATION SYSTEM 19983800 ②
REFRIG SYS 60427800 ②
INSTALLATION AND CHECKOUT 19981700 ③
INST AND CHECKOUT 60420500 ③
MAINTENANCE, PARTS 19981600 ③
MAINTENANCE, PARTS 60420600 ③
ECS SUBSYSTEM 60404700 (ECS), 60425800 (DDP), 60440500 (CONTROLLER)
M-G SET THEORY, MAINTENANCE, ETC. 60166800 (EM), 60420800 (KATO)
M-G SET DIAGRAMS 60423100 (EM), 60419900 (KATO)

NOTES:

① THIS MANUAL CONTAINS ALL SECTIONS EXCEPT MAINTENANCE, PARTS, AND INSTALLATION AND CHECKOUT.

② THIS MANUAL CONTAINS ALL SECTIONS.

③ THIS MANUAL CONTAINS INFORMATION FOR MAINFRAME AND INCLUDES ALL OPTIONS.

SYSTEM PUBLICATION INDEX

CONTENTS

FIGURES

TABLES

TYPICAL CDC CYBER 170 COMPUTER SYSTEM

SYSTEM DESCRIPTION 1

This section introduces the CDC CYBER 170 Computer Systems, gives functional characteristics of the systems, and provides descriptions of the major components of the system.

INTRODUCTION

The CDC CYBER 170 Systems include models 172 through 175. These are general-purpose digital computer systems that provide varying degrees of processing power, data storage, and input/output (I/O) capabilities.

Each system, depending upon options and design differences, includes one or more of the following major components.

- Central processor (CP)
- Central memory control (CMC)
- Central memory (CM), includes one or two central storage units (CSUs)
- Peripheral processor subsystem (PPS), includes 10 peripheral processors (PPs)
- Data channel converters (DCCs)
- Display controller
- Display station
- Condensing unit
- Extended core storage (ECS) subsystem, optional
- ECS coupler, required with optional ECS subsystem

Table 1-1 provides a comparison of the individual systems on a component level. In some systems, one or more of the components is duplicated. In such cases, manual references to the components by name or abbreviations are followed by a -0 or -1 for identification. For example, model 174 contains central processor -0 (CP-0) and central processor -1 (CP-1).

Functionally, many of the components of the systems are the same or similar. In such cases, the manual has one common description. Components that have significantly different functions have separate descriptions and are identified by the system model number.

TABLE 1-1. CDC CYBER 170 SYSTEM COMPONENTS

Components	Models			
	172	173	174	175
Mainframe:				
Central processor-0	x	x	x	-
Central processor-1	-	-	x	-
Central memory control	x	x	x	-
Central processor (includes central memory control)	-	-	-	x
Central memory, central storage unit-0	x	x	x	x
Central memory, central storage unit-1	-	*	*	x
Peripheral processor subsystem-0	x	x	x	x
Peripheral processor subsystem-1	-	*	*	*
Two data channel converters for peripheral processor subsystem-0	x	x	x	x
Two data channel converters for peripheral processor subsystem-1	-	*	*	*
Display controller	x	x	x	x
Extended core storage coupler	*	*	*	*
One 3-ton internally mounted condensing unit	x	x	–	–
Two 3-ton internally mounted condensing units	-	*	x	-
One 10-ton externally mounted condensing unit	-	-	-	x
Display station (first)	x	x	x	x
Display station (second)	*	*	*	*
Extended core storage subsystem	*	*	*	*

x = Standard
- = Not available
* = Optional

FUNCTIONAL CHARACTERISTICS

Tables 1-2 through 1-5 summarize the functional characteristics of the CP, CM, PPS, and data address and checking for each system.

TABLE 1-2. CENTRAL PROCESSOR FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Models			
	172	173	174	175
60-bit internal word	x	x	x	x
Computation in fixed-and floating-point arithmetic	x	x	x	x
Eight 60-bit operand X registers	x	x	x	x
Eight 18-bit address A registers	x	x	x	x
Eight 18-bit index B registers	x	x	x	x
Character manipulation by compare/move instructions	x	x	x	-
Synchronous internal logic with 50-nanosecond clock period	x	x	x	-
Large and small arithmetic sections	x	x	x	-
Synchronous internal logic with 25-nanosecond clock period	-	-	-	x
12-word instruction word stack	-	-	-	x
Nine functional units	-	-	-	x

x = Standard
- = Not available

TABLE 1-3. CENTRAL MEMORY FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Model			
	172	173	174	175
Cycle time of 400 nanoseconds	x	x	x	x
Maximum transfer rate of one word each 50 nanoseconds	x	x	x	x
Semiconductor memory of 32,768 words (60-bit words plus 8 error detection/correction bits per word)	x	-	-	-
Expandable to 49,152; 65,536; 98,304; and 131,072 words	*	-	-	-
Semiconductor memory of 65,536 words (60-bit words plus 8 error detection/correction bits per word)	-	x	x	x
Expandable to 98,304; 131,072; 196,608; and 262,144 words	-	*	*	*
Organized into eight independent banks per CSU	x	x	x	-
Organized into 16 independent banks throughout CSU-0 and CSU-1	-	-	-	x
x = Standard - = Not available * = Optional				

TABLE 1-4. PERIPHERAL PROCESSOR SUBSYSTEM FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Model			
	172	173	174	175
12-bit internal word	x	x	x	x
Binary computation in fixed-point arithmetic	x	x	x	x
Selectable operating speeds of 1X or 2X (1X equals major cycle of 1000 nanoseconds and minor cycle of 100 nanoseconds; 2X equals major cycle of 500 nanoseconds and minor cycle of 50 nanoseconds)	x	x	x	x
10 PPs time-share access to CM	x	x	x	x
Each PP has internal semiconductor memory of 4096 words (12-bit words plus 1 parity bit per word, odd parity)	x	x	x	x
Twelve I/O channels, each accessible by any of the PPs	x	x	x	x

TABLE 1-4. PERIPHERAL PROCESSOR SUBSYSTEM FUNCTIONAL CHARACTERISTICS (Cont'd)

Functional Characteristics	Model			
	172	173	174	175
Status and control register	x	x	x	x
Real-time clock	x	x	x	x
Each I/O channel carries 12-bit words plus 1 parity bit per word (odd parity)	x	x	x	x
Expandable to 20 PPs and 24 I/O channels	-	*	*	*

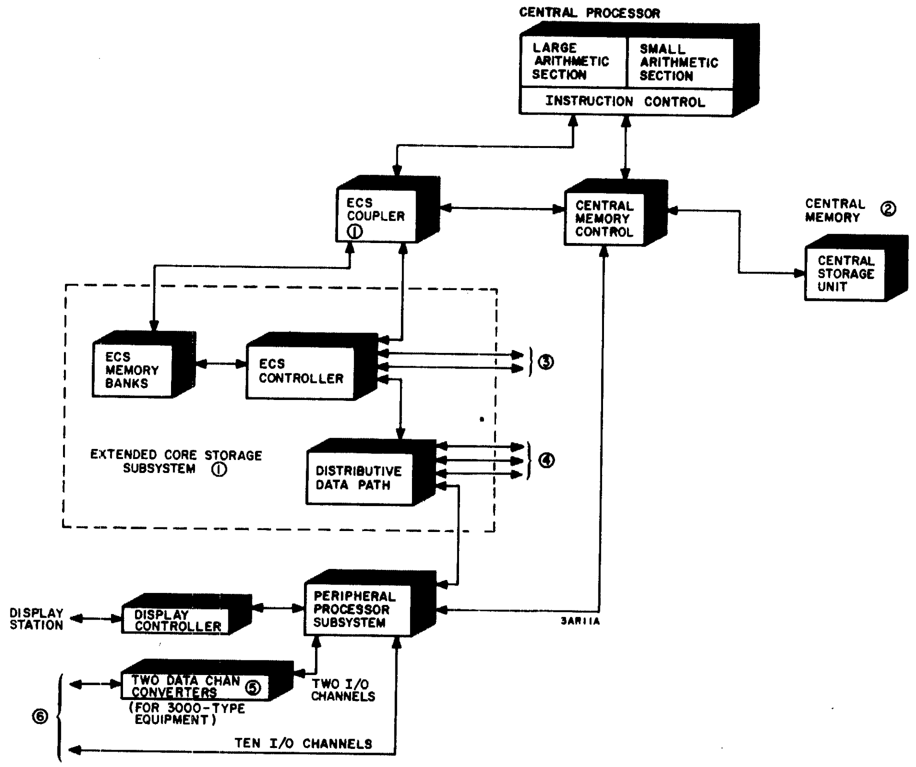
x = Standard
- = Not available
* = Optional

TABLE 1-5. DATA AND ADDRESS CHECKING FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Model			
	172	173	174	175
Parity check data between CP-0 and CMC	x	x	x	-
Parity check data between CP-1 and CMC	-	-	x	-
Parity check data between PPS-0 and CMC	x	x	x	x
Parity check data between PPS-1 and CMC	-	x	x	x
Parity check data between ECS and CMC	x	x	x	x
Single-error correction double-error detection (SECDED) between CM and CMC	x	x	x	x
Parity check address from CP-0 to CMC	x	x	x	-
Parity check address from CP-1 to CMC	-	-	x	-
Parity check address from PPS-0 to CMC	x	x	x	x
Parity check address from PPS-1 to CMC	-	x	x	x
Parity check address from CMC to CM	x	x	x	x
Parity check data between CM and CMC (non-SECDED mode only)	x	x	x	x
Parity check on PPS memory data	x	x	x	x

x = Standard
- = Not available

The model 172 basic computer system (Figure 1-1) has a serial CP that contains large and small arithmetic sections and instruction control. The CP communicates with the PPS and optional ECS only through CM which is under control of CMC. All I/O operations are performed by the PPS which uses a separate instruction set to execute independent programs in each of 10 PPs. The PPs have individual memories and can communicate with each other and any of the 12 I/O channels. When added to the system, the ECS provides additional memory capabilities, short access times, and fast transfer rates to and from CM. The model 172 also has CM options which allow expansion of CM to 131K 60-bit words.



NOTES:

① OPTIONAL EQUIPMENT.

② BASIC CM CONTAINS 32,768 60-BIT WORDS. CM IS EXPANDABLE TO 49,152; 65,536; 98,304; AND 131,072 60-BIT WORDS.

③ TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS.

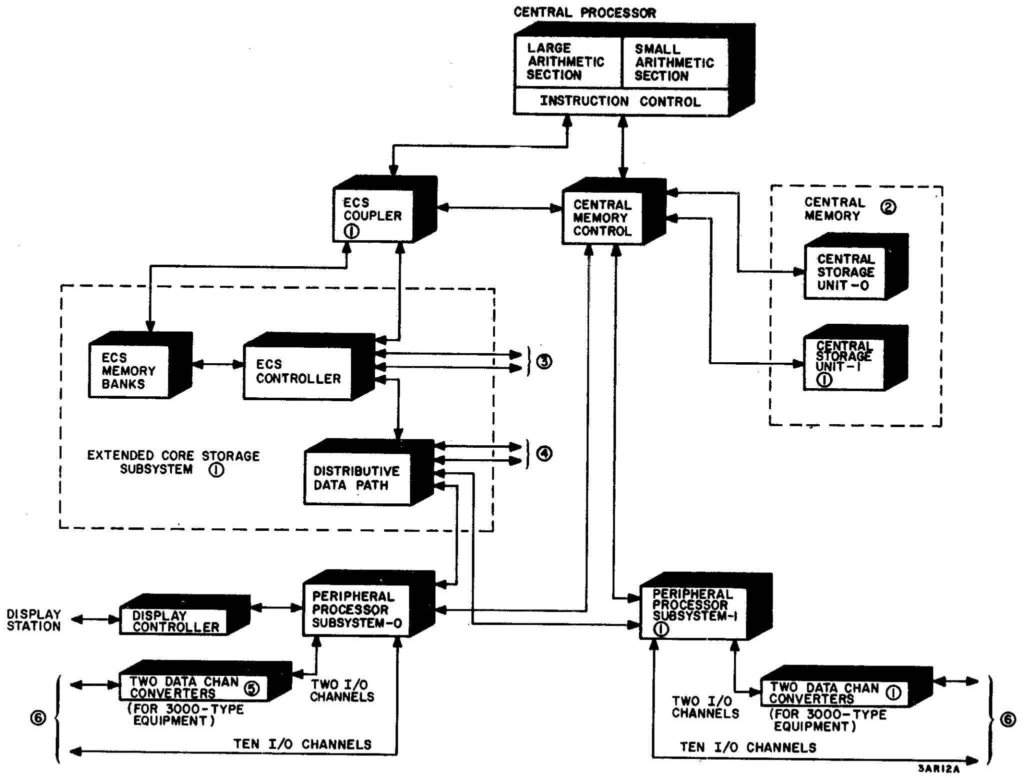
④ THREE PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

⑤ TWO DATA CHANNEL CONVERTERS ARE INCLUDED IN BASIC AND OPTIONAL PP SUBSYSTEMS. ADDITIONAL CONVERTERS MAY BE ADDED EXTERNALLY.

⑥ PERIPHERAL EQUIPMENT.

Figure 1-1. Model 172 Basic Computer System

The model 173 basic computer system (Figure 1-2) is functionally similar to the model 172, except that the CP provides faster operation, larger CM options are available, and a second PPS with additional I/O channels is available. Like the model 172, the CM may be expanded to 131K 60-bit words in CSU-0. In addition, the model 173 CM may be expanded to 262K 60-bit words by adding CSU-1. The PPS may be expanded from 10 to 14, 17, or 20 PPs by adding PPS-1. The optional PPS-1 expands the number of I/O channels from 12 to 24.



NOTES:

① OPTIONAL EQUIPMENT.

② BASIC CM CONTAINS 65,536 60-BIT WORDS. CM IS EXPANDABLE TO 98,304; 131,072; 196,608; AND 262,144 60-BIT WORDS.

③ TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS.

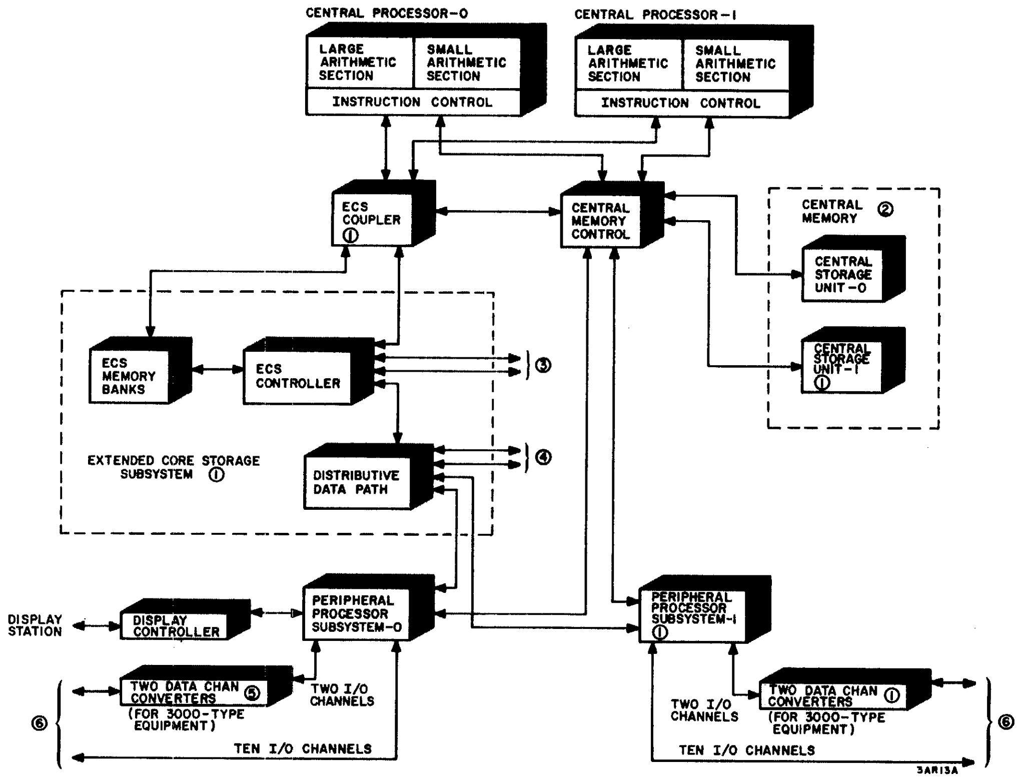
④ TWO PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

⑤ TWO DATA CHANNEL CONVERTERS ARE INCLUDED IN BASIC AND OPTIONAL PP SUBSYSTEMS. ADDITIONAL CONVERTERS MAY BE ADDED EXTERNALLY.

⑥ PERIPHERAL EQUIPMENT.

Figure 1-2. Model 173 Basic Computer System

The model 174 basic computer system (Figure 1-3) is functionally similar to model 173, except that it contains two serial CPs for dual-program processing. The ECS, CM, PPS, and I/O options are the same as for model 173.



NOTES:

① OPTIONAL EQUIPMENT.

② BASIC CM CONTAINS 65,536 60-BIT WORDS. CM IS EXPANDABLE TO 98,304; 131,072; 196,608; AND 262,144 60-BIT WORDS.

③ TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS.

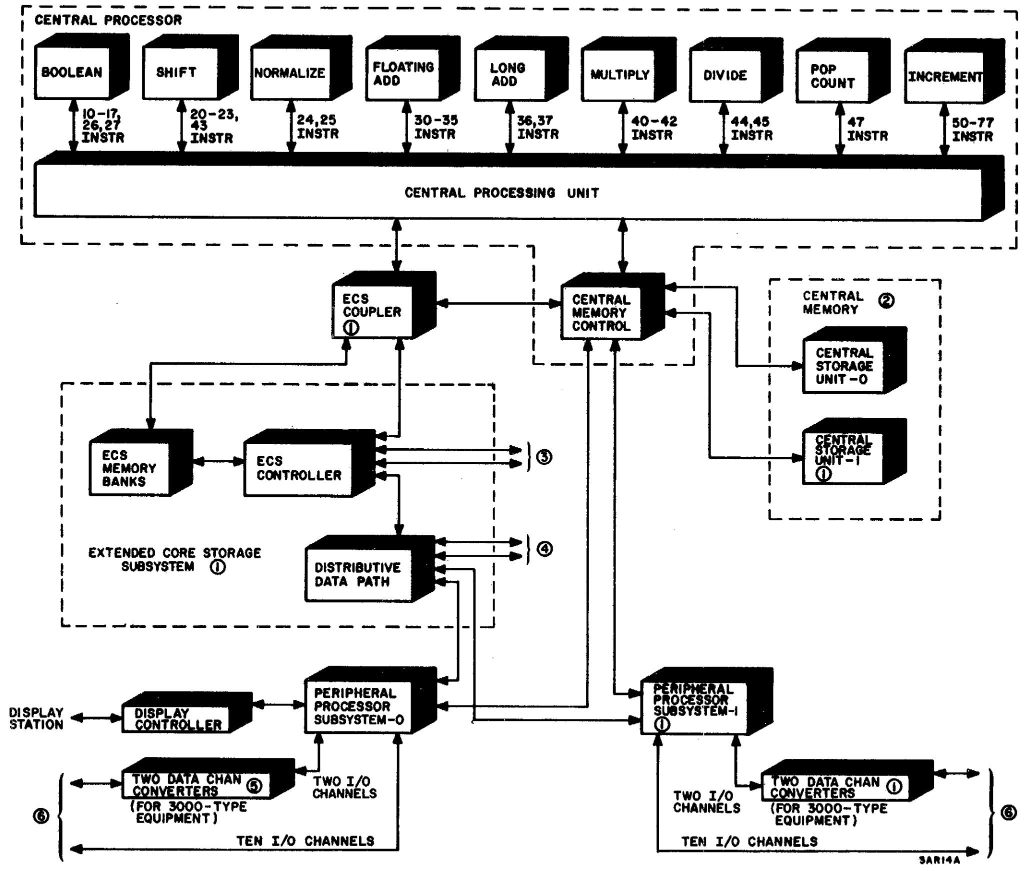
④ TWO PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

⑤ TWO DATA CHANNEL CONVERTERS ARE INCLUDED IN BASIC AND OPTIONAL PP SUBSYSTEMS. ADDITIONAL CONVERTERS MAY BE ADDED EXTERNALLY.

⑥ PERIPHERAL EQUIPMENT.

Figure 1-3. Model 174 Basic Computer System

The model 175 basic computer system (Figure 1-4) is functionally similar to model 173 and its options except in the CP. In place of the serial CP, the model 175 CP contains nine functional units, a central processing unit (CPU), and the CMC. The nine functional units operate in parallel as independent specialized arithmetic units, providing maximum overlap of instruction retrieval and execution. The basic model 175 has two CSUs, providing 16 independent banks of memory.



NOTES:

① OPTIONAL EQUIPMENT.

② BASIC CM CONTAINS 65,536 60-BIT WORDS. CM IS EXPANDABLE TO 98,304; 131,072; 196,608; AND 262,144 60-BIT WORDS.

③ TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS.

④ TWO PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

⑤ TWO DATA CHANNEL CONVERTERS ARE INCLUDED IN BASIC AND OPTIONAL PP SUBSYSTEMS. ADDITIONAL CONVERTERS MAY BE ADDED EXTERNALLY.

⑥ PERIPHERAL EQUIPMENT.

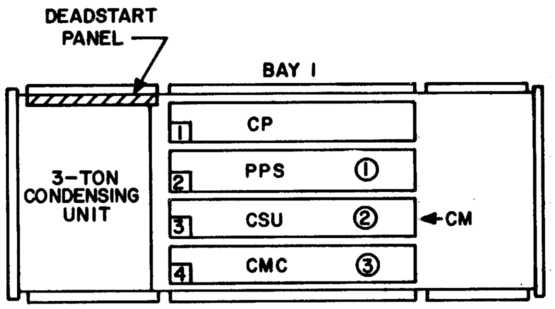
Figure 1-4. Model 175 Basic Computer System

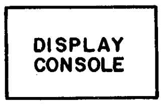
MAJOR SYSTEM COMPONENT DESCRIPTIONS

Each basic computer system includes a mainframe, display station, and internal or separately mounted condensing unit(s) as shown in Figures 1-5 through 1-8. Depending upon the system and the options, the mainframe may consist of one, two, or three bays. Optional bays may be added to the basic system as a field upgrade. The mainframe chassis, numbered 1 through 7 and 10, are accessible through hinged cabinet doors. In models 172 through 174, all of the chassis are hinged and swing out from the cabinet for maintenance. In model 175, only chassis 1 through 4 and 10 are hinged. Chassis 5 through 7 are stationary but have front and back accessibility through the cabinet doors. Additional physical characteristics are in the Site Preparation Mainframe manuals listed in the system publication index.

Conduction cooling for models 172 through 174 is provided by a 3-ton condensing unit located at one end of each mainframe bay. Conduction cooling for model 175 is provided by one 10-ton condensing unit, located as a separate unit but connected to the mainframe with liquid and suction refrigerant lines. Additional cooling information is in the Refrigeration System manuals listed in the system publication index.

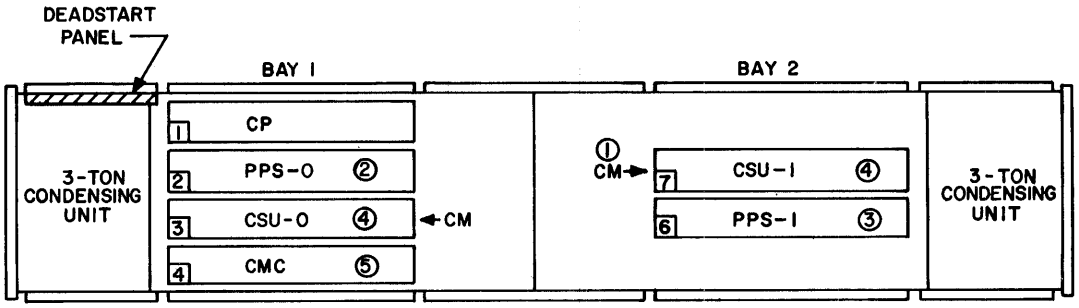
Power distribution, temperature sensing, and voltage warning sensing are handled within the mainframe. Temperature-monitor indicators and system power-control circuitry are contained in a wall-mounted panel. The dew point warning circuit is an independent wall-mounted device. Additional power and warning information is in the Power Distribution and Warning System manual listed in the system publication index.

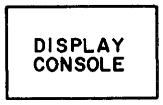




NOTES:
① CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY CONTROLLER.
② CSU IS EXPANDABLE FROM 32,768 TO 49,152; 65,536; 98,304; AND 131,072 WORDS.
③ WHEN ECS IS USED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

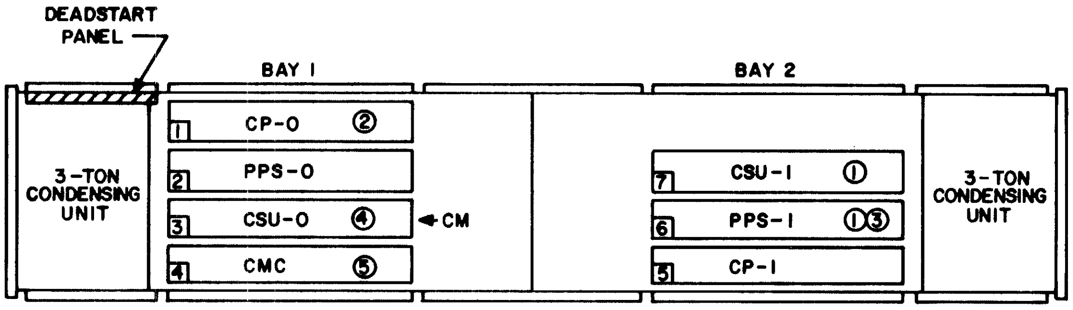
Figure 1-5. Model 172 Chassis Configuration (Top Cutaway View)

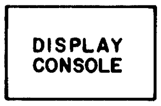




NOTES:
① BAY 2 AND CHASSIS 6 AND 7 OPTIONAL.
② CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY CONTROLLER.
③ CHASSIS 6 MAY CONTAIN TWO OPTIONAL DATA CHANNEL CONVERTERS.
④ CSU-0 IS EXPANDABLE FROM 65,536 TO 98,304 AND 131,072 60-BIT WORDS. CSU-1 IS OPTIONAL AND EXPANDABLE LIKE CSU-0. CSU-0 AND CSU-1 PROVIDE UP TO 262,144 60-BIT WORDS.
⑤ WHEN ECS IS USED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

Figure 1-6. Model 173 Chassis Configuration (Top Cutaway View)





NOTES:

① BAY 2 AND CHASSIS 6 AND 7 ARE OPTIONAL.

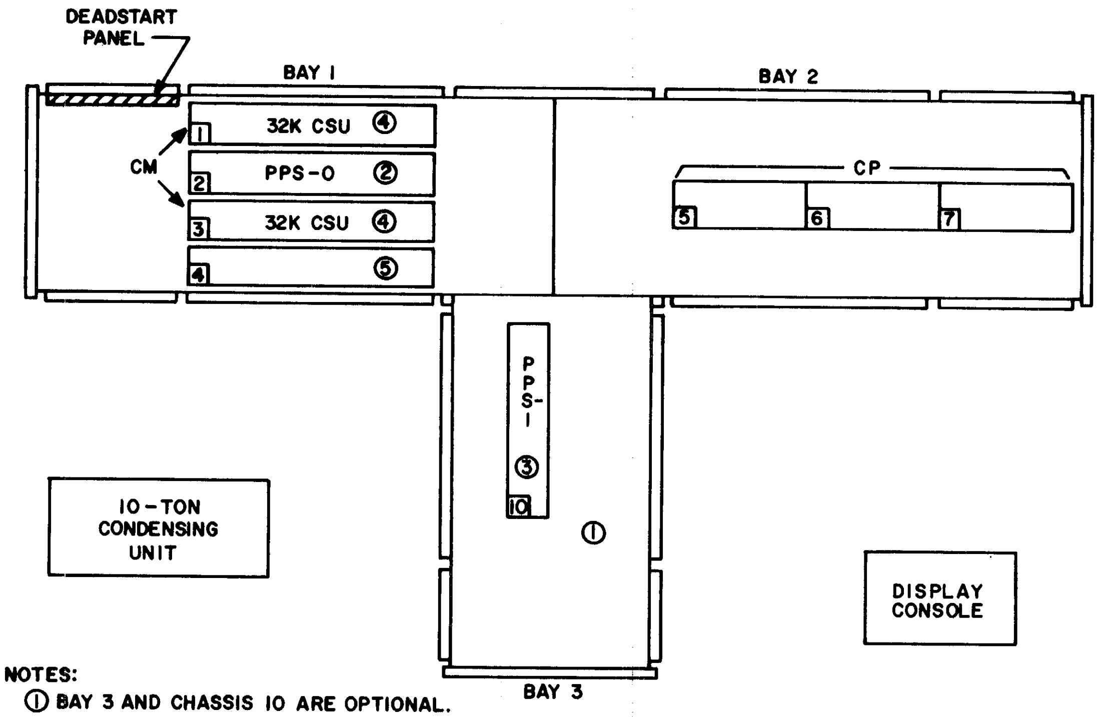
② CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY CONTROLLER.

③ CHASSIS 6 MAY CONTAIN TWO OPTIONAL DATA CHANNEL CONVERTERS.

④ CSU-0 IS EXPANDABLE FROM 65,536 TO 98,304 AND 131,072 60-BIT WORDS. CSU-1 IS OPTIONAL AND EXPANDABLE LIKE CSU-0. CSU-0 AND CSU-1 PROVIDE UP TO 262,144 60-BIT WORDS.

⑤ WHEN ECS IS USED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

Figure 1-7. Model 174 Chassis Configuration (Top Cutaway View)



NOTES:

① BAY 3 AND CHASSIS 10 ARE OPTIONAL.

② CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY CONTROLLER.

③ CHASSIS 10 MAY CONTAIN TWO OPTIONAL DATA CHANNEL CONVERTERS.

④ CSU CHASSIS ARE EXPANDABLE FROM 65,536 TO 98,304; 131,072; 196,608; 262,144 60-BIT WORDS.

⑤ WHEN ECS IS USED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

Figure 1-8. Model 175 Chassis Configuration (Top Cutaway View)

CENTRAL PROCESSOR MODELS 172 THROUGH 174

The CP consists of an instruction control section, a large arithmetic section, and a small arithmetic section. The CP is isolated from the PPS and is thus able to carry on computation or character manipulation unencumbered by I/O requirements.

The instruction control section directs the arithmetic and manipulative functions for instruction execution. The control section also performs instruction retrieving, address preparation, memory protection, and data retrieving and storing. Instructions are acquired from CM and decoded and executed in a serial manner. To reduce storage accesses for operands used during the execution of an instruction, 24 operating registers are provided. These registers are:

- Eight 60-bit X registers (X0 through X7) which hold operands used for computation.
- Eight 18-bit A registers (A0 through A7) which are CM operand address registers.
- Eight 18-bit B registers (B0 through B7) which are primarily indexing registers for controlling program execution. The B0 register always contains all zeros.

The instruction control section also contains seven support registers that support the operating registers during program execution. These registers are:

- Program address (P) register, 18 bits
- Reference address for CM (RAC) register, 18 bits
- Field length for CM (FLC) register, 18 bits
- Exit mode (EM) register, 6 bits
- Reference address for ECS (RAE) register, 21 bits
- Field length for ECS (FLE) register, 24 bits
- Monitor address (MA) register, 18 bits

The instruction control section also directs the character manipulative functions of the compare/move instructions. Characters are 6 bits, therefore, a word may contain up to 10 characters. Characters can be moved from one CM location to another, and fields of characters can be compared directly or through a collation table.

The arithmetic sections of the CP consist of a large arithmetic section (used by instructions requiring manipulation of 60-bit operands) and the small arithmetic section (used by instructions requiring manipulation of 18-bit operands). The large and small arithmetic sections also provide other arithmetic functions required by the CP for instruction execution such as instruction addressing.

CENTRAL MEMORY CONTROL MODELS 172 THROUGH 174

The CMC provides an interface between CM and five possible requesting ports (PPS-0, PPS-1 in models 173 and 174, ECS, CP-0, and CP-1 in model 174. The primary function of CMC is to provide and control address and write data to CM and read data from CM. In addition, CMC:

- Determines priorities
- Increments addresses
- Checks and generates address and data parity
- Provides single-error correction double-error detection
- Performs breakpoint checks
- Controls CM refresh
- Controls CM reconfiguration

CENTRAL PROCESSOR MODEL 175

The CP consists of a CPU, nine functional units, and a CMC. This component arrangement differs from that of the model 172 through 174 CPs which contain two arithmetic units and an instruction control. The CP differences exist because of the CP processing methods. Although the model 174 provides parallel processing by the use of two CPs, these CPs and the model 172 and 173 CPs are basically serial processors. The model 175 CP is a parallel processor. The parallel processing is performed by the nine functional units which operate independently and may operate at the same time to provide maximum concurrency of instruction execution.

The CPU basically contains 24 operating registers, 7 support registers, and instruction control.

The 24 operational registers minimize CM references for functional unit operands and results. These registers are:

- Eight 60-bit X registers (X0 through X7) which are the source and destination of operands for the functional units, input data from CM, and output data to CM
- Eight 18-bit A registers (A0 through A7) which are CM operand address registers
- Eight 18-bit B registers (B0 through B7) which are primarily indexing registers for controlling program execution

The seven support registers support the operating registers during the execution of programs. These registers are:

- P register, 18 bits
- RAC register, 18 bits
- FLC register, 18 bits
- EM register, 6 bits
- RAE register, 21 bits
- FLE register, 24 bits
- MA register, 18 bits

Instruction control basically consists of the instruction word stack (IWS), instruction address stack (IAS), current instruction word (CIW), and P registers. The IWS and IAS allow short program loops to execute without references to CM.

The nine functional units operate as independent specialized arithmetic units. These units are:

- Boolean unit which forms the logical product, logical sum, or logical difference of two 60-bit operands, transfers a 60-bit operand between X registers, and packs and unpacks floating-point operands
- Shift unit which performs mask generation and left circular or right end-off shifting of 60-bit operands

- Normalize unit which performs the normalize operation
- Floating add unit which forms the sum or difference of two floating-point operands
- Long add unit which forms the sum or difference of two 60-bit integers
- Floating multiply unit which forms the product of two floating-point operands in single or double precision and does 48-bit integer multiply
- Floating divide unit which forms the single-precision quotient of two floating-point operands
- Population count unit which counts the number of bits in a 60-bit operand which have a value of one
- Increment unit which forms the ones complement sum or difference of two 18-bit operands

Computation is performed by the functional units. Data moves into and out of the functional units through the operating registers (A, B, and X) in the CPU.

CMC controls the flow of data between CM and the requesting elements of the system. The CMC:

- Assigns priority to CM requests from the CPU, PPS-0, PPS-1, and ECS
- Resolves CM bank conflicts including bank busy and reservations for all 16 banks
- Provides control for CM read/write data
- Increments addresses for exchange jumps and ECS transfers
- Checks parity of address from PPS-0 and PPS-1
- Checks parity of data from PPS-0, PPS-1, and ECS
- Generates parity on data to PPS-0, PPS-1, and ECS
- Generates parity for address to CM in parity mode
- Generates and checks single-error correction double-error detection code bits in CM in SECDED mode
- Generates and checks data parity for CM in nonsingle-error correction double-error detection (parity or default) mode
- Checks breakpoint

CENTRAL MEMORY

CM is a metal-oxide semiconductor (MOS) memory located in CSU-0 and CSU-1, depending upon the system model and memory capacity. The basic memory of each system is field-upgradable as follows:

- Model 172 from 32,768 to 131,072 words
- Models 173 through 175 from 65,536 to 262,144 words

Each CM word contains 60 data bits and 8 single-error correction double-error detection check code bits. The words are arranged within CSU quadrants and memory banks. A CM may contain 1 to 4 quadrants with 8 or 16 memory banks per quadrant depending upon the CM requirements.

PERIPHERAL PROCESSOR SUBSYSTEM

The PPS consists of 10 functionally independent computers in PPS-0 and 4, 7, or 10 of the computers in PPS-1, when installed as an option. The computers, termed PPs, have 4096 words (12 bits plus 1 parity bit per word) of MOS memory and a repertoire of 64 instructions. The PPs share access to CM and 12 bidirectional I/O channels. The PPs are organized in a multiplexing system that allows them to share common hardware for arithmetic, logical, and I/O operations without losing speed or independence.

A status and control register is included in the PPS as a maintenance aid. This register is program-controlled and monitors error conditions that include address and data parity errors, single-error correction double-error detection conditions, and address information. Visual light displays on the PPS chassis permit monitoring some of the register status bits.

A real-time clock is included in the PPS. The clock increments once each microsecond to 4096 microseconds, then resets and continues counting.

Model 173, 174, and 175 mainframes can be expanded to 14, 17, or 20 PPs and 24 I/O channels with the addition of a second PPS chassis. The second PPS includes an abbreviated status and control register and allows accessibility of all I/O channels by all PPs.

DATA CHANNEL CONVERTERS

The DCCs are parity enhanced interfaces that permit the use of CDC 3000 Series peripheral equipment with the CDC CYBER 170 Series Computer Systems. In the basic computer systems, two DCCs are contained in the PPS-0 chassis. In an expanded system, two additional DCCs may be optionally installed in the PPS-1 chassis.

Each DCC is assigned to one I/O channel and may share the channel with up to seven other CDC 6000 or CDC CYBER Series peripheral units such as a magnetic tape controller, a console display controller, or other DCCs.

The operation of the DCC is identical to that of a CDC 6681-2 Data Channel Converter. All 6681 Data Channel Converters may be added to PPS-0 and/or PPS-1 as external equipment.

DISPLAY CONTROLLER

The controller provides digital and analog information for alphanumeric presentations on a display station screen and receives digital information from a keyboard on the display station. All operations are under program control of the PPs. The display controller is contained in the standard PPS-0 chassis. Connection to an optional second display station are available at the controller.

DISPLAY STATION

The CDC CYBER 170 Display Station provides a visual, alphanumeric readout for the computer. The receipt of symbol and position information from the computer enables displaying program information on a console 21-inch cathode-ray tube (CRT). The station also contains an alphanumeric keyboard which enables an operator to send data to the computer. The keyboard and CRT combination permits the computer operator to modify computer programs and view the result on the screen. The computer outputs two alternate, nonrelated data streams. The display station keyboard has a switch which enables the operator to select either of the data streams or to select both for presentation on the CRT. (Except for programming information in section 5, refer to the display station manual listed in the system publication index of this manual for further display station information.)

CONDENSING UNIT

A 3-ton condensing unit provides conduction cooling for each bay of models 172 through 174. The condensing unit mounts within the bay(s) as shown in Figures 1-5 through 1-7. A separate stand-alone 10-ton condensing unit provides the cooling for the model 175. Each condensing unit cools the mainframe chassis by pumping refrigerant through chassis cold bars mounted adjacent to the logic and memory modules.

EXTENDED CORE STORAGE SUBSYSTEM (OPTIONAL)

The ECS subsystem is an optional on-line, semirandom-access, magnetic-core memory system which augments CM. The subsystem has a fixed-word length and is capable of two-way communication between its memory banks and the mainframe. An ECS subsystem contains:

- ECS controller
- ECS memory banks
- Distributive data path (optional to ECS)

The ECS controller regulates access to the ECS memory banks through four access channels and can interface from one to four ECS bays. Each access channel connects to one ECS coupler or distributive data path and carries 60 data bits. One parity bit for the 60 data bits is optional, depending upon the controller. The ECS interface carries 480 data bits plus 8 parity bits. The controller performs time-sharing on the four access channels and assembles/disassembles 60-bit/480-bit words during data transfers. Data parity is generated and checked on the access channels and ECS interface.

The ECS contains 1, 2, 4, 8, or 16 memory banks, each capable of storing 131,072 60-bit words. ECS is available in sizes ranging from 131,072 words (1 bank) to 2,015,232 words (16 banks). A cabinet, termed bay, holds up to four memory banks. Each ECS bank address stores one ECS record. An ECS record contains 8 words, each consisting of 60 data bits plus 1 parity bit. References as low as one 60-bit word are possible.

The distributive data path provides a data path between ECS and the PPs. The path allows fast PP access to data in ECS using an I/O channel, and greatly reduces the data traffic through the CM.

EXTENDED CORE STORAGE COUPLER (OPTIONAL)

The ECS coupler mounts within the mainframe cabinet. The coupler interfaces the mainframe with the ECS subsystem, processing and monitoring data and control between the systems. The coupler:

- Receives initial ECS address from the CP and relays the address, request, and read or write to the ECS controller
- Receives the word count from the CP and compares the number of words transferred with the word count
- Generates and sends a continue request signal to CMC to set CM bank reservations
- Generates and sends a bank initiate signal for each transfer of a CM word
- Increments each ECS address
- Generates an end-of-transfer signal when the transfer is completed normally
- Terminates a transfer when an error condition is detected
- Provides a parity check of the word count and address information received from the CP
- Generates parity for ECS addresses transmitted to the ECS controller
- Provides a data input and output interface between CMC and the ECS controller
- Receives flag functions from the CP and relays them to the ECS controller
- Receives and sends data parity from and to CMC on model 175

FUNCTIONAL DESCRIPTIONS 2

This section describes the functions of the mainframe major components which consist of:

- Central processor (CP)
- Central memory control (CMC)
- Central memory (CM)
- Peripheral processor subsystem (PPS)
- Data channel converters (DCCs)
- Display controller

Functional information for the system display station, condensing units, and ECS is in the respective manuals, listed in the system publication index.

CENTRAL PROCESSOR MODELS 172 THROUGH 174

The CP consists of large and small arithmetic sections and an instruction control section. The arithmetic sections perform arithmetic operations by manipulation of 18- and 60-bit operands. The instruction control section directs the arithmetic operations, directs character manipulative functions of compare/move instructions, and interfaces the CMC and arithmetic sections.

ARITHMETIC SECTIONS

The large and small arithmetic sections form a unified arithmetic unit. Instructions use the large section of the unit for 60-bit operand manipulation and the small section for 18-bit operand manipulation. The large arithmetic section contains a 108-bit adder, shift network, normalize network, and shift counter. Instructions use the small arithmetic section for the manipulation of 18-bit operands and for exponent manipulation. The small arithmetic section contains an 18-bit adder. The arithmetic sections also provide other arithmetic functions required by the CP for instruction execution.

INSTRUCTION CONTROL SECTION

The instruction control section consists of 24 operating registers, 7 support registers, and logic for instruction control.

The following operating and support register descriptions are identical to those for the model 175 and are repeated in the description of the model 175 CP to provide a complete system description for that model.

Operating Registers

The operating registers consist of X, A, and B registers. These registers provide the basic function of minimizing memory references for arithmetic operands and results.

X Registers

The CP contains eight 60-bit operand (X) registers. These registers (X0 through X7) are the principal data handling registers for computation. Data flows from these registers to CM. Data also flows from CM into these registers. All 60-bit operands involved in computation must originate and terminate in these registers.

Operands and results transfer between CM and the X registers as a result of placing CM addresses into corresponding address (A) registers.

A Registers

The CP contains eight 18-bit A registers. These registers (A0 through A7) are essentially CM operand address registers associated one-for-one with the X registers. Placing a quantity into an address register (A1 through A5) causes an immediate CM read reference to that relative address and sends the CM word to the corresponding operand register (X1 through X5). Similarly, placing a quantity into address register A6 or A7 causes the word in the corresponding X6 or X7 operand register to be written into that relative address of CM.

The A0 and X0 registers operate independently of each other and have no connection with CM. They serve as intermediate results registers to be used at the user's discretion. If the system includes extended core storage (ECS), the A0 register is used during an ECS transfer to provide the relative CM starting address, while the X0 register is used to provide the relative ECS starting address in a block copy operation. The X0 register also provides the instruction information during a flag register operation. The A0 and Bj plus K registers are not used in a flag operation.

B Registers

The CP contains eight 18-bit index (B) registers. These registers (B0 through B7) are primarily indexing registers for controlling program execution. Program loop counts may be incremented or decremented in these registers.

Program addresses may be modified on the way to an A register by adding or subtracting B register quantities. The B registers also hold shift counts for the nominal (Bj) shifts, the result exponent for the unpack, the operand exponent for the pack, and the resultant shift count from a normalize. Register B0 always contains positive zero which can be used as an operand. The register cannot hold results from instructions.

Support Registers

Seven support registers are used to assist the operating registers during the execution of programs. The support registers are loaded from CM during an exchange sequence (refer to Exchange Jump in section 5). With the exception of the P register, the contents of the support registers cannot be altered during the execution interval of an exchange package. When the execution interval has been completed, the data in the support registers is sent back to CM.

P Register

An 18-bit program address (P) register is loaded from CM during the first word of an exchange sequence. The register contains the current program execution address. The register serves as a program address counter and holds the relative CM address for each program step.

RAC Register

An 18-bit reference address CM (RAC) register is loaded from CM during the second word of an exchange sequence. An absolute CM address is formed by adding RAC to the relative address that is determined by the instruction. CM references from the PPs are absolute addresses and therefore are not added to RAC. The content of the P register is added to RAC to form the absolute program address in CM. A P-equal-to-zero condition specifies relative address zero and therefore, RAC. This address is reserved for recording program-error-exit-conditions and should not be used to store data or instructions.

FLC Register

An 18-bit field length CM (FLC) register is loaded from CM during the third word of an exchange sequence. The FLC register is used to define the size of the field of the program in execution. Relative CM addresses are compared with FLC to check that the program is not going out of its allocated memory range. (For further information, refer to Central Processor Exit Mode/Error Response in section 5.)

EM Register

A six-bit exit mode (EM) register is loaded from CM during the fourth word of an exchange sequence. The EM register holds six exit mode selection bits that control individual error conditions for a program. Selected EM register bits cause the CP to error exit when the corresponding conditions occur. Any or all of the six bits can be selected at one time. Unselected EM register bits allow the CP to continue, without error processing, when most of the corresponding conditions occur. The exit mode selection bits appear in the exchange package as bits 48 through 50 and 57 through 59. The bits and their corresponding conditions are:

Mode Selection Bit	Condition Sensed
48	Address out of range
49	Operand out of range
50	Indefinite operand
57	Parity error on ECS flag register operation
58	CMC input error
59	CM data error

RAE Register

A 21-bit reference address ECS (RAE) register is loaded from CM during the fifth word of an exchange sequence. An absolute ECS address is formed by adding RAE to the relative address which is determined by the instruction.

FLE Register

A 24-bit field length ECS (FLE) register is loaded from CM during the sixth word of an exchange sequence. The FLE register is used to define the size of the field in ECS for the program in execution. Relative ECS addresses are compared with FLE. (For further information, refer to Central Processor Exit Mode/Error Response in section 5.)

MA Register

An 18-bit monitor address (MA) register is loaded from CM during the seventh word of an exchange sequence. The MA register contains the absolute starting address of an exchange package which is used when executing a central exchange jump instruction with the monitor flag clear or when honoring a monitor exchange jump to MA (262x) instruction with the monitor flag clear.

Instruction Control

The instruction control logic performs instruction translation and control sequences. Each control sequence obtains the necessary instruction operands from the operating registers and provides the control signals for execution. Instructions read from CM are 60-bit instruction words that are in four 15-bit groups, two 30-bit groups, or a combination of 15-bit and 30-bit groups. The 15-bit groups are termed parcels with the first parcel (parcel 0) being the highest-order 15 bits of a 60-bit CM word. Second, third, and fourth parcels (parcels 1, 2, and 3) follow in order. The 30-bit groups contain two 15-bit parcels.

The instruction control sequences control the execution of one or more instructions of a common type. These sequences and associated instructions are briefly described in this section. (For further information, refer to the instruction descriptions in section 4.)

Boolean Sequence

The boolean sequence controls the execution of instructions requiring bit-by-bit manipulation. This includes both the logical and transmissive operations. The instructions requiring logical operations are:

11	Logical product (Xj) and (Xk) to Xi
12	Logical sum of (Xj) and (Xk) to Xi
13	Logical difference of (Xj) and (Xk) to Xi
15	Logical product of (Xj) and ($\overline{\text{Xk}}$) to Xi
16	Logical sum of (Xj) and ($\overline{\text{Xk}}$) to Xi
17	Logical difference of (Xj) and ($\overline{\text{Xk}}$) to Xi

The instructions requiring transmissive operations are:

10	Transmit (Xj) to Xi
14	Transmit ($\overline{\text{Xk}}$) to Xi

Shift Sequence

The shift sequence controls instructions that require shifting the 60-bit field of data within the operand word. The shift instructions are:

20	Left shift (Xi) by jk
21	Right shift (Xi) by jk
22	Left shift (Xk) nominally (Bj) places to Xi
23	Right shift (Xk) nominally (Bj) places to Xi
43	Form mask of jk bits to Xi

The shift sequence also controls the pack and unpack instructions. In the packed floating format, the coefficient is contained in the lower 48 bits. The sign and biased exponent are contained in the upper 12 bits. The unpack instruction obtains the packed word from the Xk register, delivers the coefficient to the Xi register, and delivers the exponent to the Bj register. The unpack and pack instructions are:

26	Unpack (Xk) to Xi and Bj
27	Pack (Xk) and (Bj) to Xi

The shift sequence also controls the normalize operations. The coefficient portion of the operand is repositioned and the exponent is adjusted so that the most significant bit of the coefficient is in the highest-order bit position of the coefficient, and the exponent is decreased by the number of bit positions shifted. The normalize instructions are:

24	Normalize (Xk) to Xi and Bj
25	Round normalize (Xk) to Xi and Bj

Floating-Add Sequence

The floating-add sequence controls the operations necessary to form the 48-bit floating sum with a 12-bit exponent of the floating-point sum or difference of two floating-point operands. The floating-add instructions are:

30	Floating sum of (Xj) and (Xk) to Xi
31	Floating difference of (Xj) and (Xk) to Xi
32	Floating double-precision sum of (Xj) and (Xk) to Xi
33	Floating double-precision difference of (Xj) and (Xk) to Xi
34	Round floating sum of (Xj) and (Xk) to Xi
35	Round floating difference of (Xj) and (Xk) to Xi

Floating-Multiply and-Divide Sequence

The floating multiply and -divide sequence controls the operation of floating-multiply, -divide, and population-count instructions.

The multiply instructions are:

40	Floating product of (Xj) and (Xk) to Xi
41	Rounding floating product of (Xj) and (Xk) to Xi
42	Floating double-precision product of (Xj) and (Xk) to Xi

The divide instructions are:

44	Floating divide (Xj) by (Xk) to Xi
45	Round floating divide (Xj) to (Xk) to Xi

The population-count instruction counts the number of one bits in a 60-bit operand. The instruction is:

47	Population count of (Xk) to Xi

Increment Sequence

The increment sequence controls the ones complement addition and subtraction of 18-bit fixed-point operands for increment instructions 50 through 77. The sequence also controls the 60-bit ones complement sum and difference values for long add instructions 36 and 37.

The increment instructions are:

50	Set Ai to (Aj) + K
51	Set Ai to (Bj) + K
52	Set Ai to (Xj) + K
53	Set Ai to (Xj) + (Bk)
54	Set Ai to (Aj) + (Bk)
55	Set Ai to (Aj) - (Bk)
56	Set Ai to (Bj) + (Bk)
57	Set Ai to (Bj) - (Bk)
60	Set Bi to (Aj) + K
61	Set Bi to (Bj) + K
62	Set Bi to (Xj) + K
63	Set Bi to (Xj) + (Bk)
64	Set Bi to (Aj) + (Bk)
65	Set Bi to (Aj) - (Bk)
66	Set Bi to (Bj) + (Bk)
67	Set Bi to (Bj) - (Bk)
70	Set Xi to (Aj) + K
71	Set Xi to (Bj) + K
72	Set Xi to (Xj) + K
73	Set Xi to (Xj) + (Bk)
74	Set Xi to (Aj) + (Bk)
75	Set Xi to (Aj) - (Bk)
76	Set Xi to (Bj) + (Bk)
77	Set Xi to (Bj) - (Bk)

The long add instructions are:

36	Integer sum of (Xj) and (Xk) to Xi
37	Integer difference of (Xj) and (Xk) to Xi

Compare/Move Sequence

The compare/move sequence controls the execution of compare/move instructions that handle data manipulation on a character basis. The compare/move instructions use four support registers for source and result field CM addresses and character position offsets. The compare/move instructions are:

464	Move indirect (Bj + K)
465	Move direct
466	Compare collated
467	Compare uncollated

The four support registers are:

An 18-bit K1 register which specifies which CM word contains the first character of the source data field.

An 18-bit K2 register which specifies which CM word contains the first character of the result field.

A 4-bit C1 register which specifies the character position or offset of the first CM word of the source field.

A 4-bit C2 register which specifies the character position or offset of the first CM word of the result field.

Exchange Sequence

The exchange sequence generates timed CM reference signals to implement the exchange of data between the CP and CM, as required by the exchange jump instruction. In addition, the exchange sequence provides internal control signals to the operating and control registers to systematically enter the contents of an exchange jump package.

The exchange sequence is always initiated by the CMC from a CP or PP request.

ECS Block Copy Sequence

The ECS block copy sequence controls the transfer of data between CM and ECS. The number of words to be transferred is determined by the addition of K to the content of Bj. The starting address for CM is obtained from the A0 register plus the RAC reference address. The starting address for ECS is obtained from the X0 register plus the RAE reference address. The ECS block copy instructions are:

011	Block copy (Bj + K) from ECS to CM
012	Block copy (Bj + K) from CM to ECS

Normal Jump Sequence

The normal jump sequence controls the execution of branch instructions 02 through 07. The 02 instruction performs an unconditional jump to the Bi register address plus K. The branch address is K when i equals 0. The 02 instruction is:

02	Jump to (Bk) + K

The conditional jump instructions 03 through 07 branch to address K if the jump condition is met. These instructions are:

030	Branch to K if (Xj) = 0
031	Branch to K if (Xj) $\neq$ 0
032	Branch to K if (Xj) positive
033	Branch to K if (Xj) negative
034	Branch to K if (Xj) in range
035	Branch to K if (Xj) out of range
036	Branch to K if (Xj) definite
037	Branch to K if (Xj) indefinite
04	Branch to K if (Bi) = (Bj)
05	Branch to K if (Bi) $\neq$ (Bj)
06	Branch to K if (Bi) $\geq$ (Bj)
07	Branch to K if (Bi) < (Bj)

Return Jump Sequence

The return jump sequence controls the execution of two instructions, which are:

00	Error exit to MA or program stop
010	Return jump to K

CENTRAL MEMORY CONTROL MODELS 172 THROUGH 175

The CMC controls the flow of data between CM and the requesting system components. In models 172 through 174, CMC is separated from the CP chassis. In model 175, CMC is part of the CP chassis. The model 175 CMC location eliminates the need for interchassis address and data parity checks from the CP and the need for CMC to generate data parity to the CP. The CMC:

- Assigns priority to CM requests from:

  CP, PPS, and ECS (model 172)
  CP, PPS-0, PPS-1, and ECS (model 173)
  CP-0, CP-1, PPS-0, PPS-1, and ECS (model 174)
  CP, PPS-0, PPS-1, and ECS (model 175)

- Resolves CM bank conflicts including bank busy and reservations for the 8 or 16 CM banks

- Provides control for CM read/write data

- Increments addresses for exchange jumps and ECS transfers

- Controls data transfers, during an exchange jump, between:

  CM and CP (models 172 and 173)
  CM and CP-0, CP-1 (model 174)
  CM and CP (model 175)

- Parity checks addresses from:

  CP and PPS (model 172)
  CP, PPS-0, and PPS-1 (model 173)
  CP-0, CP-1, PPS-0, and PPS-1 (model 174)
  PPS-0 and PPS-1 (model 175)

- Parity checks data from:

  CP, PPS, and ECS (model 172)
  CP, PPS-0, PPS-1, and ECS (model 173)
  CP-0, CP-1, PPS-0, PPS-1, and ECS (model 174)
  PPS-0, PPS-1, and ECS (model 175)

- Generates parity on data to:

  CP, PPS, and ECS (model 172)
  CP, PPS-0, PPS-1, and ECS (model 173)
  CP-0, CP-1, PPS-0, PPS-1, and ECS (model 174)
  PPS-0, PPS-1, and ECS (model 175)

- Generates parity for address to CM
- Generates and checks data parity for CM in parity mode
- Breakpoint checks
- Confronts CM refresh
- Controls CM reconfiguration
- Performs single-error correction double-error detection (SECDED) on each memory word in SECDED mode, described in Single-Error Correction Double-Error Detection in this section

CENTRAL MEMORY REFERENCE PRIORITIES

When a CM reference is initiated in models 172 through 174, the address goes to all CM banks. Only one bank is selected and accepts the address. If the bank is busy, the address is held in an address buffer until the bank is not busy. The next address (in case of a CP reference to CM) is not issued until CM accepts the reference from the address buffer. Because the model 174 has a second CP, references from the second CP may be issued and received by CM banks that are not busy. When the two CPs issue CM references at the same time to a common bank, CP-0 has priority over CP-1.

When a CM reference is initiated in model 175, the address goes to all CM banks. Only one bank is selected and accepts the address. If the bank is busy, the request waits in a storage address stack (SAS) until that bank is free. If the two-word SAS is full or a backup condition (rank A and rank B full) exists, instruction issue for instructions 50 through 57 stops. Thus, requests for two addresses may be waiting in the SAS at the same time. Instruction issue does not start again until all unaccepted addresses, up to two, have been accepted by CM. This address backup condition in SAS does not occur when doing an ECS transfer.

In all CDC CYBER 170 models, all addresses presented to CM are processed in the order in which they are received. CM requests received simultaneously are given a priority that determines which address is allowed access first. These priorities are:

- CP exchange sequence (CEJ) request for CP-0, then CP-1 for model 174; CEJ request for PPS-0, then PPS-1 for models 173 through 175
- Exchange sequence request
- ECS block copy
- I/O request for PPU-0, then PPU-1
- Return jump request for CP-0, then CP-1 (model 174)
- Increment unit request for CP-0, then CP-1 (model 174)
- Instruction fetch request for CP-0, then CP-1 (model 174)

All memory references appear the same to CM. The hardware provides tags that identify the source or destination of any CM word referenced.

CMC contains 16 bank busy registers and 16 corresponding reservation registers. The bank busy registers are set by a go signal sent to the corresponding bank. The reservation registers are set during an ECS transfer to ensure that the required banks will be free when needed for ECS.

SINGLE-ERROR CORRECTION DOUBLE-ERROR DETECTION

Single-error correction double-error detection (SECDED) is a normal CMC operating mode that permits unimpeded computer operation despite a single-bit CM failure. This mode is accomplished by a SECDED network which corrects single data errors from CM. (The SECDED mode is manually set with a switch that also allows the CMC to be set in a non-SECDED or parity mode.) In the parity mode, the SECDED network is bypassed to permit testing of the noncorrected data by writing uncoded data and reading it back through the disabled correcting network. In case of a SECDED logic failure, parity mode may be selected to continue processing after a system reload.

In the SECDED mode, the SECDED network (Figure 2-1) affects all CM write and read operations. In a write operation, a SECDED code generator sends 8 SECDED code bits with each 60 bits of write data to CM. In a read operation, CM sends the 60 read data and 8 SECDED code bits to the SECDED network. A logical zero network forces the upper 4 bits (60 through 63) of the CM read data to constant zeros. These zeros are unused but are required to satisfy the SECDED algorithm that requires 64 data bits. The 64 data bits and 8 SECDED code bits enter a read data holding register. The holding register sends the 64 data bits to a single-error correction network and the 72 data and SECDED bits to a syndrome code generator. The generator forms an eight-bit syndrome code.

When a single data bit fails, a syndrome code containing three or five bits is generated. The single-error correction network automatically corrects the incorrect bit. If two data bits fail, a syndrome code containing an even number of bits is generated. No correction is made and a double-error signal is sent to the status and control register and requesting port. The requesting port also receives a transmission parity bit for each data word read. If a multiple error occurs, the single-error correction network treats a resulting syndrome code (containing an even number of bits) as a double error. A resulting syndrome code with an odd number of bits is treated as a single error. However, some combinations of multiple-bit failures may result in a legitimate single-error syndrome code. This would result in complementing a bit that may or may not have been correct. Table 2-1 lists the octal codes for all the combinations of syndrome bits with the number of the data bit assigned each code or a note categorizing the code.

When there is no bit failure, the read data passes through the single-error correction network unchanged. All read data from the single-error correction network contains the forced zeros in the upper four bits. These bits are unused and terminated. The remaining 60 data bits go to the requesting ports.

The eight syndrome and six address bits associated with the memory reference are sent to the status and control register. This information can then be interpreted to determine the failing CSU, memory bank, memory quadrant, and failing bit (in the case of single correctable errors). This information makes it possible for maintenance personnel to isolate the failure to a module level.

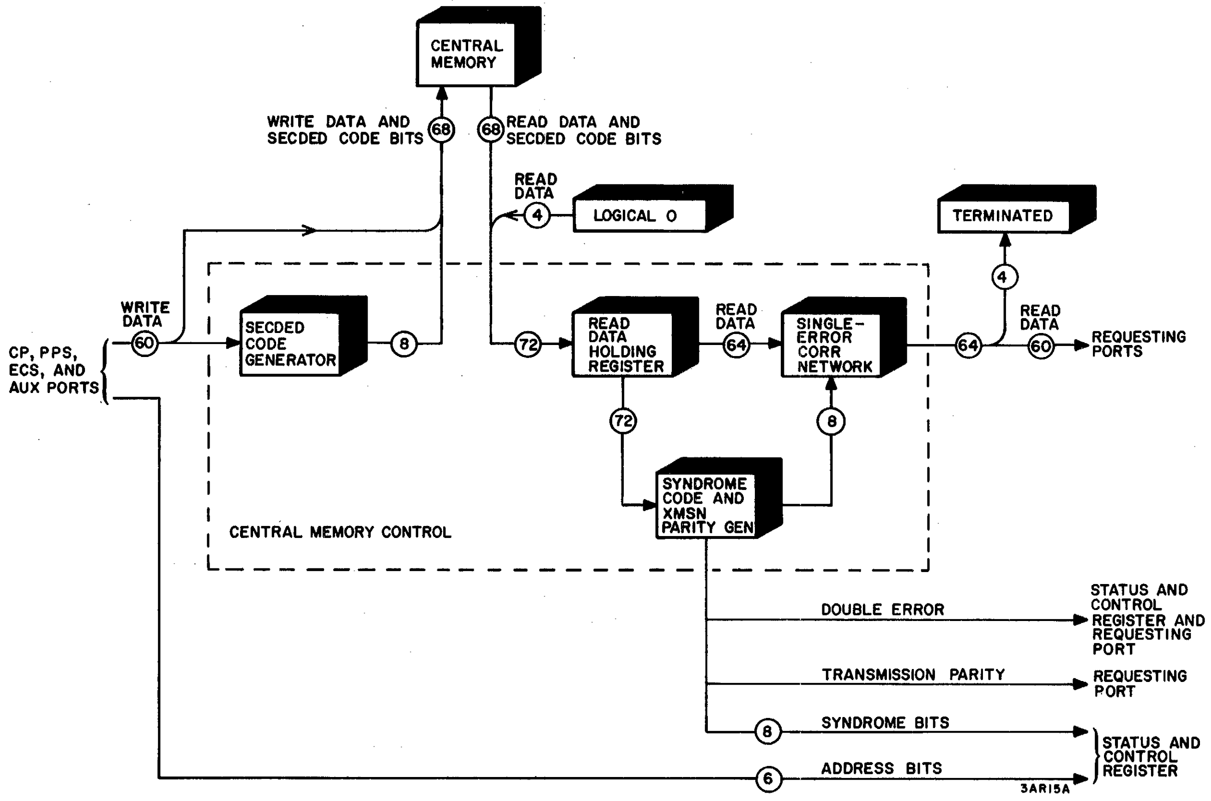

CENTRAL MEMORY
WRITE DATA AND SECDED CODE BITS
68
68
READ DATA AND SECDED CODE BITS
READ DATA
4
LOGICAL 0
TERMINATED
4
WRITE DATA
60
CP, PPS, ECS, AND AUX PORTS
SECDED CODE GENERATOR
8
72
READ DATA HOLDING REGISTER
READ DATA
64
SINGLE-ERROR CORR NETWORK
64
READ DATA
60
REQUESTING PORTS
72
8
SYNDROME CODE AND XMSN PARITY GEN
CENTRAL MEMORY CONTROL
DOUBLE ERROR
STATUS AND CONTROL REGISTER AND REQUESTING PORT
TRANSMISSION PARITY
REQUESTING PORT
8
SYNDROME BITS
6
ADDRESS BITS
STATUS AND CONTROL REGISTER
3AR15A

Figure 2-1. SECDED Network Block Diagram (SECDED Mode)

ERROR DETECTION AND RESPONSE

CMC checks for address parity errors, data parity errors, SECDED errors, and breakpoint conditions. When errors occur or breakpoint conditions are met, information is sent to the status and control register and to the requesting port. Refer to Figure 2-2 for all CMC error communications.

TABLE 2-1. SECDED SYNDROME CODES/CORRECTED BITS

CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT
000	⑤	040	①	100	①	140	②	200	①	240	②	300	②	340	50
001	①	041	②	101	②	141	53	201	②	241	57	301	58	341	②
002	①	042	②	102	②	142	54	202	②	242	59	302	④	342	②
003	②	043	0	103	1	143	②	203	2	243	②	303	②	343	③
004	①	044	②	104	②	144	40	204	②	244	④	304	④	344	②
005	②	045	23	105	3	145	②	205	5	245	②	305	②	345	③
006	②	046	22	106	8	146	②	206	9	246	②	306	②	346	③
007	10	047	②	107	②	147	③	207	②	247	44	307	③	347	②
010	①	050	②	110	②	150	41	210	②	250	43	310	48	350	②
011	②	051	47	111	7	151	②	211	6	251	②	311	②	351	28
012	②	052	27	112	31	152	②	212	11	252	②	312	②	352	③
013	13	053	②	113	②	153	③	213	②	253	③	313	③	353	②
014	②	054	29	114	30	154	②	214	16	254	②	314	②	354	③
015	17	055	②	115	②	155	③	215	②	255	③	315	③	355	②
016	18	056	②	116	②	156	③	216	②	256	③	316	③	356	②
017	②	057	③	117	52	157	②	217	③	257	②	317	②	357	③
020	①	060	②	120	②	160	42	220	②	260	45	320	49	360	②
021	②	061	46	121	51	161	②	221	56	261	②	321	②	361	③
022	②	062	32	122	55	162	②	222	15	262	②	322	②	362	③
023	14	063	②	123	②	163	③	223	②	263	③	323	36	363	②
024	②	064	33	124	35	164	②	224	39	264	②	324	②	364	20
025	19	065	②	125	②	165	③	225	②	265	③	325	③	365	②
026	21	066	②	126	②	166	③	226	②	266	③	326	③	366	②
027	②	067	③	127	③	167	②	227	③	267	②	327	②	367	③
030	②	070	34	130	37	170	②	230	38	270	②	330	②	370	③
031	24	071	②	131	②	171	③	231	②	271	③	331	③	371	②
032	25	072	②	132	②	172	12	232	②	272	③	332	③	372	②
033	②	073	③	133	③	173	②	233	③	273	②	333	②	373	③
034	26	074	②	134	②	174	③	234	②	274	③	334	③	374	②
035	②	075	4	135	③	175	②	235	③	275	②	335	②	375	③
036	②	076	③	136	③	176	②	236	④	276	②	336	②	376	③
037	③	077	②	137	②	177	③	237	②	277	③	337	③	377	②

3AR3C

NOTES

Syndrome codes are octal representations of eight syndrome code bits. Circled numbers in the bit columns refer to the following:

① Syndrome code bit failed (single code bit set).

② Double error or multiple error (even number of code bits set).

③ Multiple error reported as single error (five or seven code bits set).

④ Not used because of 64-bit algorithm.

⑤ No error was detected.

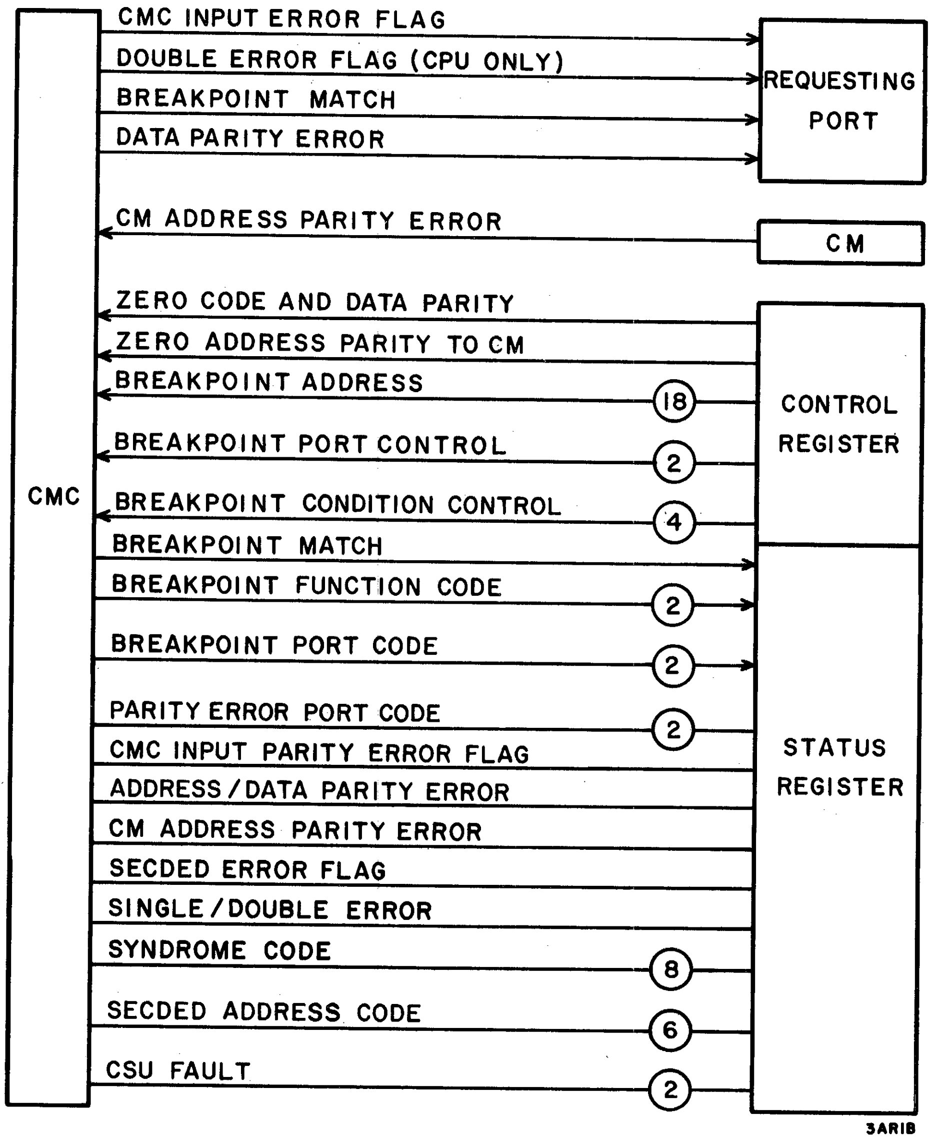


Figure 2-2. CMC Error Communications

ADDRESS PARITY

The CMC checks parity on the address paths from:

- CP and PPS (model 172)
- CP, PPS-0, and PPS-1 (model 173)
- CP-0, CP-1, PPS-0, and PPS-1 (model 174)
- PPS-0 and PPS-1 (model 175)

If an address parity error occurs at the CMC, applicable error information is sent to the status and control register as follows:

- CMC input parity error flag
- Parity error port code
- Address error versus data error

If the address parity error occurs on a write request, the write signal is blocked (not sent to CM) to protect memory.

If the address parity error occurs on a read request, the read data is replaced by a word of all ones.

Address parity is generated in CMC for the address going to CM. If CM detects an error, the error signal is sent back to CMC. The CMC then sends applicable error information to the status and control register as follows:

- CSU-0 error
- CSU-1 error

If CP is the requesting port to CM, the CMC input error flag signal sets the parity error condition. If the exit mode bit 58 is set, additional action is taken in the CP. Refer to Central Processor Exit Mode/Error Response in section 5 for further information.

DATA PARITY

The CMC checks parity on the data paths from:

- PPS-0 to CMC
- PPS-1 to CMC (models 173, 174, and 175)
- ECS to CMC
- CP to CMC (models 172 and 173)
- CP-0 and CP-1 to CMC (model 174)

If a data parity error occurs at the CMC, a CMC input error signal is sent to the requesting port which initiated the reference, and applicable error information is sent to the status and control register as follows:

- Parity error flag
- Requesting port code or ECS error flag

The previous signals are the same as the address parity information with the exception of the address error. The absence of the address error indicates a data error. Refer to Status and Control Register in section 5 for additional parity information.

In a parity mode on a write operation, the data parity in models 172 through 175 is generated in the CMC for transmission to CM and substituted in place of SECDED code bit 10.

In parity mode on a read operation, the data parity bit in models 172 through 174 is propagated (unchanged) for interrogation by the requesting unit. In model 175, parity is checked on the data from CM and code bit 0 is used as the parity bit. When a parity error occurs in model 175, the CMC sends only an error signal to the central processing unit (CPU) if the CPU is the requesting unit. For other ports in model 175, the parity bit is propagated for interrogation by the requesting unit.

In SECDED mode on a write operation, data parity is checked at the input requesting ports of all models (except the CPU port in model 175). SECDED code bits are then generated for transmission to CM.

In SECDED mode on a read operation, all models send data through the SECDED network. A parity bit is then generated in CMC and transmitted to the requesting unit (except the CPU in model 175) along with the read data.

BREAKPOINT CHECK

The CMC performs a breakpoint check on references to CM when breakpoint is selected. Breakpoint is controlled by the status and control register in the PPS.

The CMC receives 18 breakpoint address bits, 2 port control bits, and 2 access control bits. Table 2-2 lists the breakpoint control translations.

The 18-bit address of each CM reference is compared to the breakpoint address bits. If there is a match, if the requesting unit is selected by the port control bits, and if the type of access is one that is selected by the access control bits, the breakpoint flag is sent to the requesting unit.

The breakpoint flag is also sent to the status and control register along with the two port code bits. For further information, refer to Breakpoint in section 5.

When executing an exchange jump, this operation is treated by breakpoint as both a read and a write. A return jump is treated as a write.

TABLE 2-2. BREAKPOINT CONTROL TRANSLATIONS

Control Bit				Translation
117	116	115	114	
0	0	X	X	Breakpoint check disabled
0	1	X	X	Breakpoint check for PP ports
1	0	X	X	Breakpoint check for CP ports
1	1	X	X	Breakpoint check for PP and CP ports
X	X	0	0	Breakpoint check on read
X	X	0	1	Breakpoint check on write
X	X	1	0	Breakpoint check on read next instruction
X	X	1	1	Breakpoint check on any access

CENTRAL PROCESSOR MODEL 175

The CP consists of a CPU, nine functional units, and a CMC. The CPU, together with the functional units, executes programs stored in CM. The CMC controls all CM accesses.

The CMC of the model 175 is located within the CP chassis of bay 2, unlike the CMCs of models 172 through 174 which are located in individual chassis of bay 1. Although the physical locations of the CMCs of some models differ, the CMC functions of all models are similar and are therefore previously described in Central Memory Control in section 2.

CENTRAL PROCESSING UNIT

The CPU consists of 24 operating registers, 7 support registers, and instruction control. The CPU includes the registers and control logic to direct the arithmetic operations and provide interface between the functional units and CMC. In addition to instruction execution, the CPU performs instruction fetching, address preparation, memory protection, and data fetching and storing. Figure 2-3 illustrates the general flow of information.

Program execution begins with execution of an exchange jump which loads the CPU operating registers from a 16-word block from CM. Exchange jump also stores the original contents of the CPU operating registers in the same 16-word block in CM. The operating system can use an exchange jump to switch program execution between two CM programs, leaving the first program in a usable state for later reentry. (For further information, refer to Exchange Jump in section 5.)

The CPU reads 60-bit words from CM and enters them in the instruction word stack (IWS) which is capable of holding up to 12 60-bit words. Each instruction word in turn leaves the IWS and enters the current instruction word (CIW) register for interpretation and testing. The CIW register holds four 15-bit instructions, two 30-bit instructions, or combinations of the two types of instructions. The 15-bit or 30-bit instructions issue individually from the CIW register. The functional units obtain the instruction operands from and store results in 24 operating registers. Reservation control keeps an account of active operating registers to resolve conflicts.

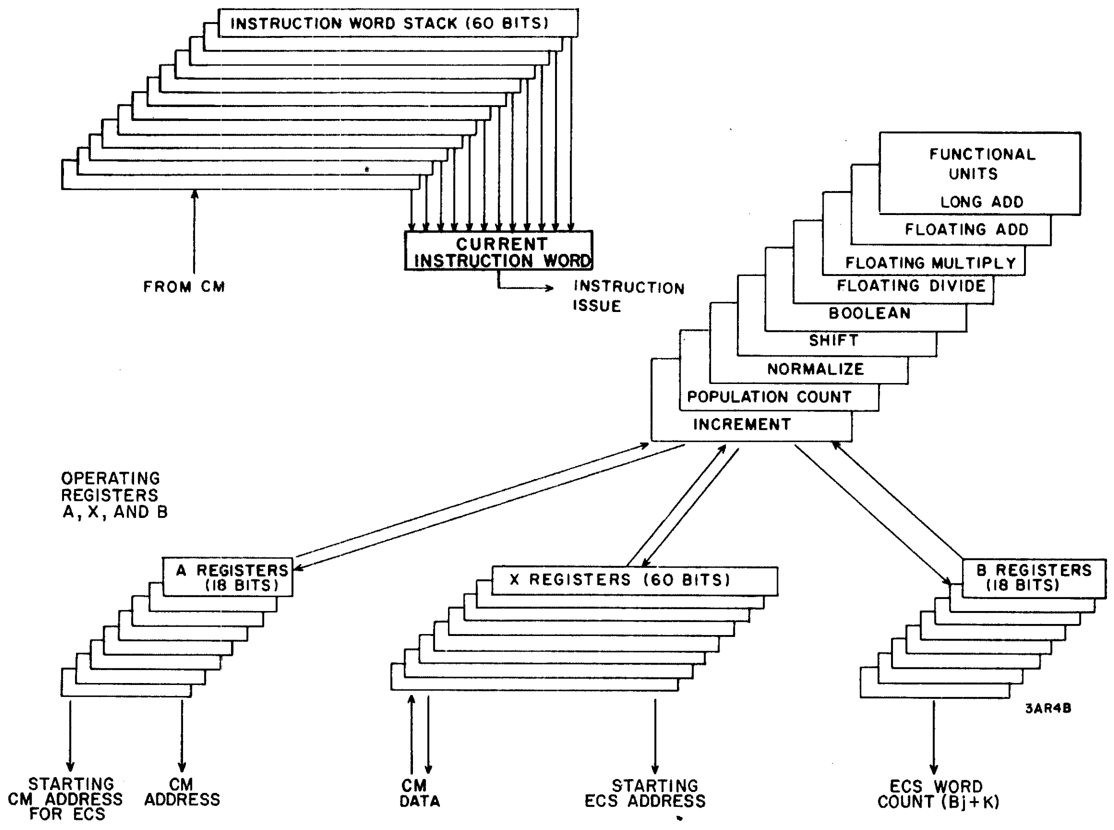


Figure 2-3. Model 175 CPU Information Flow

The following operating and support register descriptions, although identical to those for models 172 through 174, are repeated here to provide a complete system description.

Operating Registers

The operating registers consist of X, A, and B registers. These registers minimize memory references for arithmetic operands and results.

X Registers

The CPU contains eight 60-bit operand (X) registers. These registers (X0 through X7) are the principal data handling registers for computation. Data flows from these registers to CM. Data also flows from CM into these registers. All 60-bit operands involved in computation must originate and terminate in these registers. Operands and results transfer between CM and the X registers as a result of placing CM addresses into corresponding address registers.

A Registers

The CPU contains eight 18-bit address (A) registers. These registers (A0 through A7) are CM operand address registers associated one-for-one with the X registers. Placing a quantity into an address register (A1 through A5) causes an immediate CM read reference to that relative address and sends the CM word to the corresponding operand register (X1 through X5). Similarly, placing a quantity into address register A6 or A7 causes the word in the corresponding X6 or X7 operand register to be written into that relative address of CM.

The A0 and X0 registers operate independently of each other and have no connection with CM. They serve as intermediate results registers to be used at the user's discretion. If the system includes ECS, the A0 register is used during an ECS transfer to provide the relative CM starting address, while the X0 register is used to provide the relative ECS starting address for a read or write operation. The X0 register also provides the instruction information during a flag register operation. The A0 and Bj plus K registers are not used in a flag operation.

B Registers

The CPU contains eight 18-bit index (B) registers. These registers (B0 through B7) are primarily indexing registers for controlling program execution. Program loop counts may be incremented or decremented in these registers.

Program addresses may be modified on the way to an A register by adding or subtracting B register quantities. The B registers also hold shift counts for the nominal Bj shifts, the result exponent for the unpack, the operand exponent for the pack, and the resultant shift count from a normalize. Register B0 always contains positive zero which can be used as an operand. The register cannot hold results from instructions.

Support Registers

Seven support registers are used to assist the operating registers during the execution of programs. The support registers are loaded from CM during an exchange sequence (refer to Exchange Jump in section 5). With the exception of the P register, the contents of the support registers cannot be altered during the execution interval of an exchange package. When the execution interval has been completed, the data in these registers is sent back to CM.

P Register

An 18-bit program address (P) register is loaded from CM during the first word of an exchange sequence. The register contains the current program execution address. The register serves as a program address counter and holds the relative CM address for each program step.

RAC Register

An 18-bit reference address CM (RAC) register is loaded from CM during the second word of an exchange sequence. An absolute CM address is formed by adding RAC to the relative address that is determined by the instruction. CM references from the PPs are absolute addresses and therefore are not added to RAC. The content of the P register is added to RAC to form the absolute program address in CM. A P-equal-to-zero condition specifies relative address zero and therefore RAC. This address is reserved for recording program-error-exit-conditions and should not be used to store data or instructions.

FLC Register

An 18-bit field length CM (FLC) register is loaded from CM during the third word of an exchange sequence. The FLC register is used to define the size of the field of the program in execution. Relative CM addresses are compared with FLC to check that the program is not going out of its allocated CM range. (For further information, refer to Central Processor Exit Mode/Error Response in section 5.)

EM Register

A six-bit exit mode (EM) register is loaded from CM during the fourth word of an exchange sequence. The EM register holds six exit mode selection bits that control individual error conditions for a program. Selected EM register bits cause the CP to error exit when the corresponding conditions occur. Any or all of the six bits can be selected at one time. Unselected EM register bits allow the CP to continue, without error processing, when most of the corresponding conditions occur. The exit mode selection bits appear in the exchange package as bits 48 through 50 and 57 through 59. The bits and their corresponding conditions are:

Mode Selection Bit	Condition Sensed
48	Address out of range
49	Operand out of range
50	Indefinite operand
57	Parity error on ECS flag register operation
58	CMC input error
59	CM data error

RAE Register

A 21-bit reference address ECS (RAE) register is loaded from CM during the fifth word of an exchange sequence. An absolute ECS address is formed by adding RAE to the relative address that is determined by the instruction.

FLE Register

A 24-bit field length ECS (FLE) register is loaded from CM during the sixth word of an exchange sequence. The FLE register is used to define the size of the field in ECS for the program in execution. Relative ECS addresses are compared with FLE. (For further information, refer to Central Processor Exit Mode/Error Response in section 5.)

MA Register

An 18-bit monitor address (MA) register is loaded from CM during the seventh word of an exchange sequence. The MA register contains the absolute starting address of an exchange package which is used when executing a central exchange jump instruction with the monitor flag clear or when honoring a monitor exchange jump to MA (262x) instruction with the monitor flag clear.

Instruction Control

The main instruction control components include an IWS, IAS, CIW, and P register. The instruction control reads 60-bit instruction words from CM and issues them in 15-, 30-, or 60-bit instruction groups to the CP functional units for execution. The instruction control also performs instruction translation and control of the exchange, ECS block copy, normal jump, and return jump sequences.

Instruction Word Stack

The IWS is a group of 12 60-bit registers that hold program instruction words for execution. It is essentially a moving window in the program code. The IWS is filled two words ahead of the program address currently being executed. A program loop of up to 10 instruction words may be entirely contained within the IWS. When this happens, the instruction loop may be executed repeatedly without further references to CM. When an instruction causes a jump out of stack, the stack is voided. Voiding the stack means that the IWS is not accessible, and the IAS is cleared. New instructions must then be read from CM into the IWS and the IAS.

The 12 IWS registers are individually identified by rank. The rank 1 register contains the oldest data. If the IWS contains sequential program instruction words, the content of the rank 1 register corresponds to the lowest CM address in the IAS.

The IWS is shifted to accommodate a new word arriving from CM. New information arriving from CM is entered in rank 12. Ranks 11 through 1 are cleared and entered with information from the next highest-order rank. The information in rank 1 is discarded.

Instruction Address Stack

The IAS is a group of 12 18-bit address registers associated with the IWS. It holds relative CM program addresses on a one-for-one basis with the program words in the IWS. The rank 1 register contains the CM address from which the word in rank 1 of the IWS is read. All ranks are compared with the current program address. If coincidence occurs for a rank, the corresponding rank in the IWS is sent to the 60-bit CIW register.

A maintenance switch is provided to disable either IAS ranks 1 through 10 or ranks 1 through 4.

Current Instruction Word Register

The CIW register is divided into four 15-bit parcels. All four parcels are loaded when an instruction word is read from the IWS. An instruction issues from the CIW register when conditions in the functional units and operating registers are such that the instruction will be executed without conflicting with previously issued instructions. The other parcels are then left shifted in the CIW register by either 15 or 30 bits, depending upon the instruction format.

Program Address Register

The 18-bit P register contains the current program execution address. The register serves as a program address counter and holds the relative CM address for each program step. Since the P register is advanced one address ahead of the instruction in progress, a P register holds the current program execution address. This buffered address is used for the PPS read program instruction. The content of the P register advances to the next program step as follows:

1. The P register is advanced by one when an instruction word is sent to the CIW register.

2. The P register is set to the address specified by a branch instruction. If the instruction is a return jump, the current P plus 1 is stored before entering P with the new value to allow a return to the original sequence.
3. The P register is set to the address specified in the exchange package.

Exchange Sequence

The exchange sequence generates timed CM reference signals to implement the exchange of data between the CP and CM, as required by the exchange jump instruction. In addition, the exchange sequence provides internal control signals to the operating and control registers to systematically enter the contents of an exchange jump package.

The exchange sequence is always initiated by the CMC from a CP or PP request.

ECS Block Copy Sequence

The ECS block copy sequence controls the transfer of data between CM and ECS. The number of words to be transferred is determined by the addition of K to the value in Bj. The starting address for CM is obtained from the A0 register plus the RAC reference address. The starting address for ECS is obtained from the X0 register plus the RAE reference address. The ECS block copy instructions are:

011	Block copy (Bj + K) from ECS to CM
012	Block copy (Bj + K) from CM to ECS

Normal Jump Sequence

The normal jump sequence controls the execution of branch instructions 02 through 07. The 02 instruction performs an unconditional jump to the Bi register address plus K, causing the instruction stack to be voided. The branch address is K when i equals 0. The 02 instruction is:

02	Jump to (Bi) + K

The conditional jump instructions 03 through 07 branch to address K if the jump condition is met. These instructions are:

030	Branch to K if (Xj) = 0
031	Branch to K if (Xj) ≠ 0
032	Branch to K if (Xj) positive
033	Branch to K if (Xj) negative
034	Branch to K if (Xj) in range
035	Branch to K if (Xj) out of range
036	Branch to K if (Xj) definite
037	Branch to K if (Xj) indefinite
04	Branch to K if (Bi) = (Bj)
05	Branch to K if (Bi) ≠ (Bj)
06	Branch to K if (Bi) ≥ (Bj)
07	Branch to K if (Bi) < (Bj)

Return Jump Sequence

The return jump sequence controls the execution of two instructions which are:

00	Error exit to MA or program stop
010	Return jump to K

FUNCTIONAL UNITS

Each of the nine functional units in the CP is a specialized arithmetic unit with algorithms for a portion of the CP instructions. Each unit is independent of the other units, and a number of functional units may be in operation at the same time. No visible registers are in the functional units from a programming standpoint. A functional unit receives one or two operands from operating registers at the beginning of instruction execution and delivers the result to the operating registers when the function has been performed. No information is retained in a functional unit for reference in subsequent instructions.

All functional units, with the exception of the floating-multiply and -divide units, have a 1-clock-period segmentation. This means that the information arriving at a unit, or moving within a unit, is captured and held in a new set of registers every clock period. Therefore, it is possible to start a new set of operands for unrelated computation in a functional unit each clock period even though the unit may require more than 1 clock period to complete the calculation. This process may be compared to a delay line in which data moves through the unit in segments to arrive at the destination in the proper order but at a later time. All functional units perform their algorithms in a fixed amount of time. No delays are possible once the instruction has issued.

The floating-multiply unit has a 2-clock-period segmentation. Operands may enter the multiply unit in any clock period providing there was no multiply instruction initiated in the preceding clock period. There is a 1-clock-period delay in initiating a multiply instruction if another multiply instruction has just started.

The floating-divide unit is the only functional unit in which an iterative algorithm is executed. There is no segmentation possible in this unit although the beginning of a new operation can overlap the completion of the previous operation by 2 clock periods.

Boolean Unit

The boolean unit executes instructions which require bit-by-bit data manipulation. This includes both the logical operations and the transmissive operations. The instructions requiring logical operations are:

11	Logical product (Xj) and (Xk) to Xi
12	Logical sum of (Xj) and (Xk) to Xi
13	Logical difference of (Xj) and (Xk) to Xi
15	Logical product of (Xj) and ($\overline{\text{Xk}}$) to Xi
16	Logical sum of (Xj) and ($\overline{\text{Xk}}$) to Xi
17	Logical difference of (Xj) and ($\overline{\text{Xk}}$) to Xi

The instructions requiring transmissive operations are:

10	Transmit (Xj) to Xi
14	Transmit ($\overline{\text{Xk}}$) to Xi
26	Unpack (Xk) to Xi and Bj
27	Pack (Xk) and (Bj) to Xi

Shift Unit

The shift unit executes instructions which require shifting the entire 60-bit field of data within the operand word. The applicable instructions are:

20	Left shift (Xk) to jk
21	Right shift (Xi) to jk
22	Left shift (Xk) nominally (Bj) places to Xi
23	Right shift (Xk) nominally (Bj) places to Xi
43	Form mask of jk bits to Xi

Normalize Unit

The normalize unit executes instructions which require rearranging operands in floating-point format. The unit left shifts the coefficient so that the most significant bit is shifted into the highest-order bit position of the coefficient. The exponent is adjusted by subtracting the shift count. The applicable instructions are:

24	Normalize (Xk) to Xi and Bj
25	Round normalize (Xk) to Xi and Bj

Floating-Add Unit

The floating-add unit executes instructions which require adding operands in floating-point format. The applicable instructions are:

30	Floating sum of (Xj) and (Xk) to Xi
31	Floating difference of (Xj) and (Xk) to Xi
32	Floating double-precision sum of (Xj) and (Xk) to Xi
33	Floating double-precision difference of (Xj) and (Xk) to Xi
34	Round floating sum of (Xj) and (Xk) to Xi
35	Round floating difference of (Xj) and (Xk) to Xi

Long Add Unit

The long add unit executes instructions which require integer addition of two 60-bit operands. The applicable instructions are:

36	Integer sum of (Xj) and (Xk) to Xi
37	Integer difference of (Xj) and (Xk) to Xi

Multiply Unit

The multiply unit executes instructions which require multiplication of two operands in floating-point format. The applicable instructions are:

40	Floating product of (Xj) and (Xk) to Xi
41	Rounded floating product of (Xj) and (Xk) to Xi
42	Floating double-precision product of (Xj) and (Xk) to Xi

Divide Unit

The divide unit executes instructions which require division of two operands in floating-point format. The applicable instructions are:

44	Floating divide (Xj) by (Xk) to Xi
45	Round floating divide (Xj) by (Xk) to Xi

Population-Count Unit

The population count unit executes an instruction which requires counting the number of one bits in a 60-bit operand. The applicable instruction is:

47	Population count of (Xk) to Xi

Increment Unit

The increment unit executes instructions (50 through 77) which require arithmetic operations on two 18-bit operands. During the 50 through 57 instructions, the result is transmitted to an A register. The same result plus RAC is sent to CM for the increment read or write address. The two operations are performed independently and in parallel with each other. During 60 through 67 instructions, the result is transmitted to a B register. During 70 through 77 instructions, the result is transmitted to an X register.

The increment instructions are:

50	Set Ai to (Aj) + K
51	Set Ai to (Bj) + K
52	Set Ai to (Xj) + K
53	Set Ai to (Xj) + (Bk)
54	Set Ai to (Aj) + (Bk)
55	Set Ai to (Aj) - (Bk)
56	Set Ai to (Bj) + (Bk)
57	Set Ai to (Bj) - (Bk)
60	Set Bi to (Aj) + K
61	Set Bi to (Bj) + K
62	Set Bi to (Xj) + K
63	Set Bi to (Xj) + (Bk)
64	Set Bi to (Aj) + (Bk)
65	Set Bi to (Aj) - (Bk)
66	Set Bi to (Bj) + (Bk)
67	Set Bi to (Bj) - (Bk)
70	Set Xi to (Aj) + K
71	Set Xi to (Bj) + K
72	Set Xi to (Xj) + K
73	Set Xi to (Xj) + (Bk)
74	Set Xi to (Aj) + (Bk)
75	Set Xi to (Aj) - (Bk)
76	Set Xi to (Bj) + (Bk)
77	Set Xi to (Bj) - (Bk)

CENTRAL MEMORY

Each model of the computer system has basic and optional CM sizes. The CM sizes are determined by the number of 68-bit words, 60 data bits and 8 SECDED bits, that the CMs store as shown in Tables 2-3 and 2-4. The basic CM size for each system is the smallest size listed for that system. The optional sizes are the next four larger sizes.

A CM consists of CSU-0 and CSU-1, depending upon model and CM options. Each CSU contains eight CM banks. The banks are numbered 0 through 7 in each CSU of models 172 through 174. In model 175, which always contains CSU-0 and CSU-1, the banks are numbered 0 through 15. The number of words contained in a CM bank depends upon the CM size which is determined by the number of quadrants within each CSU.

A quadrant is a division of CM that contains eight CM banks in CSU-0. When CSU-1 is used, the quadrant includes the additional CM banks in that unit. Up to three quadrants may be added to increase any of the basic CM sizes. The addition of quadrants causes the words per CM bank to increase. For example, the words per bank in model 172 increase from 4096 to 16,384 with the addition of quadrants 1 through 3. A special application of quadrant 1 in models 172 and 175 permits a CM size to be increased without the addition of a complete quadrant. Quadrants are added with plug-in CM modules that contain semiconductor memory chips.

The CMs have phased addressing which consists of a sequential bank addressing and sequential word addressing. Sequential address references from CMC to the CMs may occur each 50 nanoseconds (maximum rate). This rate and a CM cycle time of 400 nanoseconds permit up to eight CM banks to be active at any one time. Each CM bank has an access time of 400 nanoseconds at the CSU chassis ports and a maximum data transfer rate of one word each 50 nanoseconds.

TABLE 2-3. MODELS 172 THROUGH 174 CENTRAL MEMORY SIZES

System Model	CM Size (Words)	Words Per Bank	CSU-0 Memory Banks 0 1 2 3 4 5 6 7	CSU-1 Memory Banks 0 1 2 3 4 5 6 7
172	32,768	4,096	Quadrant 0	
172	49,152	6,144	Quadrant 0	
			Quadrant 1	
172				
173	65,536	8,192	Quadrant 0	
174			Quadrant 1	
172			Quadrant 0	
173	98,304	12,288	Quadrant 1	
174			Quadrant 2	
172			Quadrant 0	
173	131,072	16,384	Quadrant 1	
174			Quadrant 2	
			Quadrant 3	
		16,384	Quadrant 0	Quadrant 0
173		in CSU-0	Quadrant 1	Quadrant 1
174	196,608	8,192	Quadrant 2	
		in CSU-1	Quadrant 3	
173			Quadrant 0	Quadrant 0
174	262,144	16,384	Quadrant 1	Quadrant 1
			Quadrant 2	Quadrant 2
			Quadrant 3	Quadrant 3

TABLE 2-4. MODEL 175 CENTRAL MEMORY SIZES

System Model	CM Size (Words)	Words Per Bank	CSU-0 Memory Banks 0, 1, 2, 3, 4, 5, 6, 7 — CSU-1 Memory Banks 8, 9, 10, 11, 12, 13, 14, 15
175	65,536	4,096	Quadrant 0
175	98,304	6,144	Quadrant 0
			Quadrant 1
175	131,072	8,192	Quadrant 0
			Quadrant 1
175	196,608	12,288	Quadrant 0
			Quadrant 1
			Quadrant 2
175	262,144	16,384	Quadrant 0
			Quadrant 1
			Quadrant 2
			Quadrant 3

ADDRESS FORMAT

The location of each word in CM is identified by an 18-bit address in CMC. The format for the address and a resulting CSU address format for models 172 through 174 are shown in Figure 2-4, and for model 175 in Figure 2-5. The two CMC address formats differ because of addressing requirements within the models. The differences exist in the location of the CSU SEL bit. In models 172 through 174, the bit is in position 17. In model 175, the bit is in position 3. Each CMC address format provides 14 bits for the CSU address format. This format is the same in all models.

The CMC address format bits address one CSU word by first selecting CSU-0 or CSU-1 with the CSU SEL bit and one of the eight CM banks within the selected CSU with the BANK SEL bits. The CM word is further addressed by the QUAD SEL bits which select one of four quadrants, narrowing the word selection to one bank and one quadrant. The CHIP SEL bits select one of four columns of semiconductor memory chips on the CM modules in the selected bank and quadrant. At this point, 68 memory chips are selected. Each chip is capable of storing 1024 bits. One bit is selected from each of the 68 chips by the CELL ADDRESS bits to complete the addressing of one 68-bit word.

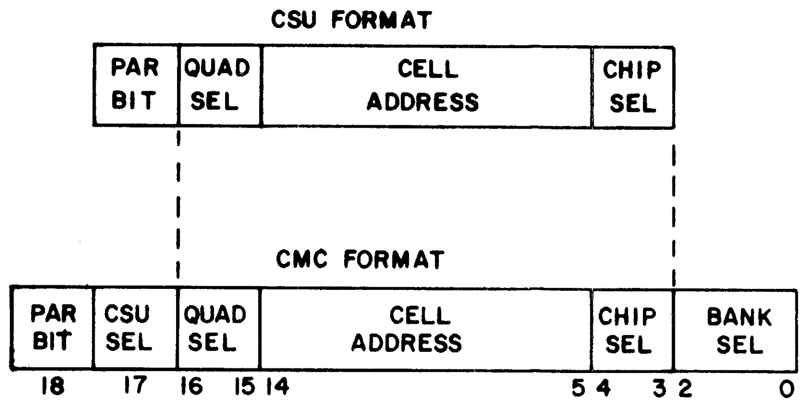


Figure 2-4. Models 172 through 174 CM Address Formats

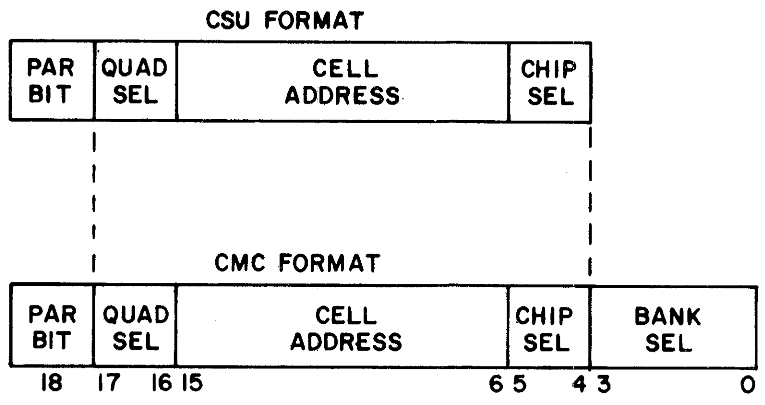


Figure 2-5. Model 175 CM Address Formats

ADDRESS PARITY

CM accepts the 14-bit address from CMC with 1 parity bit. Address parity is checked and an error signal is sent to CMC if a parity error is detected. If the address parity error occurs, a write operation is blocked within CM to protect memory, and a read operation is blocked (returning all ones) to maintain user security.

CENTRAL MEMORY REFERENCE OPERATIONS

The three major operations of CM are read, write, and refresh. These operations are under CMC control.

During a read or write CM reference, CMC sends the address information to CM. The CM sends the address information to all banks. A go bank signal from CMC, decoded from the bank select code, is sent to one of the banks. Only the bank receiving the go bank signal gates the address and data (write operation) into holding registers. The holding registers then select the storage locations and place the data into CM. During a read operation, the addressing is the same except that the absence of a write signal causes data to be read from the addressed location and sent to a common data-out register for transmission to CMC.

CM refresh is, in effect, a CM read operation. A refresh fully restores capacitive charges to storage-bit cells within the semiconductor memory chips of CM. The chips need to be refreshed because they tend to lose capacitive charges that determine whether ones or zeros are stored in the chips. To recharge the chips, CMC sends a go refresh signal to each bank of CM every 25.6 microseconds, providing that an ECS transfer or an exchange jump is not in progress. Each go refresh signal initiates a 400-nanosecond refresh cycle. Each refresh cycle accesses one-fourth of the entire CM. Each memory chip requires 32 refresh cycles (32 addresses) to completely refresh the chip. Therefore, refreshing the entire CM requires 128 (4 times 32) refresh cycles. With the refresh cycles occuring every 25.6 microseconds, the entire CM can be refreshed every 3.2768 milliseconds.

During an ECS block transfer, a refresh cycle cannot be started for all the banks at the same time without affecting the ECS transfer. To avoid degradation of ECS transfer rate, the go refresh signal is issued selectively to a phased bank which would not be referenced for the next 400 nanoseconds during the ECS block transfer.

CM refresh waits for an exchange jump to complete before initiating one of the cycles.

CENTRAL MEMORY RECONFIGURATION

Central memory reconfiguration is a manually performed function that permits the computer operator to restructure the CM addresses so that a failing part of CM can be quickly locked out to provide a continuous block of usable CM. CM reconfiguration is accomplished by setting switches to manipulate upper address bits. Hardware configures the CM quadrants so that sequential addressing is maintained. Reconfiguration options for the four models are:

Model 172	131K to 98K to 65K to 32K and 49K to 32K
Model 173	262K to 196K to 131K to 98K to 65K
Model 174	262K to 196K to 131K to 98K to 65K
Model 175	262K to 196K to 131K to 65K and 98K to 65K (only if 32K portion fails)

A reconfiguration permits only one part of the CM to be locked out at a time. The reconfiguration provides the same-sequential addressing characteristics as a same-size normally operating CM without reconfiguration. CM reconfiguration switching information is described in Central Processor Chassis Model 175 and Central Memory Control Chassis Models 172 through 174 in section 5.

REFRESH FAULT

Each CSU checks for a refresh fault which is caused by constant refresh or a refresh which may occur too often and cause excessive power dissipation. Upon detection of a refresh fault, a CM disable flip-flop sets and prevents any request or refresh access to CM. Until the CM disable flip-flop clears, the CSU returns all ones on a CM read. A CSU fault status bit sets in the status and control register. A master clear is required to clear the CM disabled flip-flops. While the CM disable flip-flop is set, CM cannot be refreshed; therefore, data is no longer valid and must be reloaded after the fault condition is cleared.

PERIPHERAL PROCESSOR SUBSYSTEM

The PPS consists of 10 PPs. Each PP is a functionally independent computer that has a memory and a repertoire of 64 instructions. The PPs share access to CM and 12 bidirectional I/O channels. The PPs are organized in a multiplexing system, termed barrel and slot, which allows them to share common hardware for arithmetic, logical, and I/O operations without losing speed or independence.

The PPS can be expanded to 14, 17, or 20 PPs in models 173 through 175 systems. The expansion is accomplished by adding a second PPS chassis to the mainframe and always includes an expansion of 12 additional I/O channels. Any of the PPs can access any of the I/O channels.

The PPS operates in a 1000-nanosecond (1X mode) major cycle time for CDC CYBER 70 compatibility or in a 500-nanosecond (2X mode) major cycle time. The major cycle time is selectable with bit 84 of the status and control register. All PPs communicate with either external equipment or each other over the 12 or 24 independent (12-bits plus 1 parity bit) bidirectional I/O channels. Only one piece of external equipment can communicate over one channel at a time, but all channels can be active at the same time.

Channel instructions direct all activities with external equipment. These instructions are used to select any equipment on any channel and transfer data to or from the selected equipment.

Each PP exchanges data with CM through CMC in 60-bit words. In a write operation, five successive 12-bit PP words are assembled into a 60-bit word and sent to CMC. In a read operation, a 60-bit word from CM is disassembled into five 12-bit words and sent to successive locations in the peripheral processor memory (PPM). Separate assembly read and write paths to CM are time-shared by each of the 10 PPs. Assembly/disassembly registers are called pyramids. These pyramids are also provided for the 4, 7, or 10 PPs in the second PPS chassis.

Data transmission parity is generated on all CM writes and sent to CMC. Data transmission parity is checked on all CM reads. If a data parity error is detected, a bit is set in the status and control register.

DEADSTART

Deadstart is a PPS operation that provides initial starting of the computer, dumping of the contents of peripheral and control processor memories to an output device (normally a printer), or sweeping PPMs without executing instructions. Deadstart is controlled by the deadstart panel in bay 1. The panel includes controls for selecting any PP as logical PP-0 (control PP). Another control enables central exchange jump/monitor exchange jump (CEJ/MEJ). (For further information, refer to Exchange Jump in section 5.)

PERIPHERAL PROCESSOR MEMORY

Each PP has an independent 4096-word, 13-bit (12 data bits plus 1 parity bit) MOS memory.

PPM data words are checked for parity on each read. If a parity error is detected, a bit is set in the status and control register. All PPs of a PPS can be selected to stop on PPM parity error by setting bit 95 in the status and control register.

A PPM reconfiguration feature permits the user to restore the PPS operation after a critical failure of a PPM designated as PPM-0. PPM-0 has a special controlling function at deadstart time. The reconfiguration is accomplished by logically exchanging the failing PPM with a good PPM and degrading the PPS to operate without the failing PPM. Logical selection of PPM-0 is performed by setting switches on the deadstart panel. Degrading the PPS must be done through the operating system. The reconfiguration permits computer operation to continue without the failing PPM so that the failing PPM can be corrected during scheduled maintenance.

The PPMs use the same memory chips as CM and also require refreshing. Refreshing occurs once each 32 microseconds and requires a period of 500 nanoseconds. During PPS operation at a 1X speed, the refresh cycle is invisible to PP operations. At a 2X speed, all PPM references are locked out during a refresh cycle.

BARREL AND SLOT

The 10 PPs are combined in a multiplexing arrangement termed barrel and slot (Figure 2-6). This arrangement allows the accumulator (A), program address (P), auxiliary accumulator (Q), and translation (K) registers of each PP to time-share common instruction-control hardware. The hardware-sharing permits logical, I/O, and other PP operations to occur without sacrificing speed or independence of the individual PPs. The barrel and slot arrangement includes common data paths to and from CM and to and from 12 I/O channels.

The barrel is a matrix of flip-flops that holds the current instruction and operand for each of nine PPs while the slot contains the current instruction and operand for the tenth PP. The barrel gives each PP a turn at using the common instruction-control hardware in the slot by shifting the quantities around the barrel from the slot output to the slot input.

Each time data enters the slot, a portion of the instruction for that data is executed. The slot performs tasks such as arithmetic and logic operations and program address manipulation. Complete execution of an instruction may require the A, P, Q, and K register quantities to go more than one trip around the barrel and through the slot.

The barrel and slot operation provides PP program operating speeds of 1X and 2X. These speeds are program-selectable by a single bit in the status and control register. When cleared, the bit causes the PPs to operate at 1X speed that consists of a 50-nanosecond slot time for each PP once every 1000 nanoseconds. When set, the bit causes the PPs to operate at 2X speed that consists of a 50-nanosecond slot time for each PP once every 500 nanoseconds.

The PPM may be referenced once each time the PP passes around the barrel and through the slot. During its slot time, the PP may also communicate in 12-bit quantities with CM or with any of the I/O channels except during a CM refresh cycle.

The 12-bit quantities that go to CM are assembled into 60-bit words before being transferred. Similarly, the 60-bit words from CM are disassembled into 12-bit quantities prior to use in the barrel and slot.

The PPMs are numbered 0 through 9. PPS SELECT switches on the deadstart panel permit any one of the PPMs to be selected as PPM-0. Following a PPM-0 selection, the 10 PPMs remain in order with new but consecutive numbers that follow PPM-0.

For example, if PPM-8 is selected as PPM-0, then PPM-9 becomes PPM-1 and the original PPM-0 becomes PPM-2. Since each PPM is associated with a PP, the same numbering scheme applies to the PPs. Only the PP numbering changes. The PPs retain the same positions in the barrel.

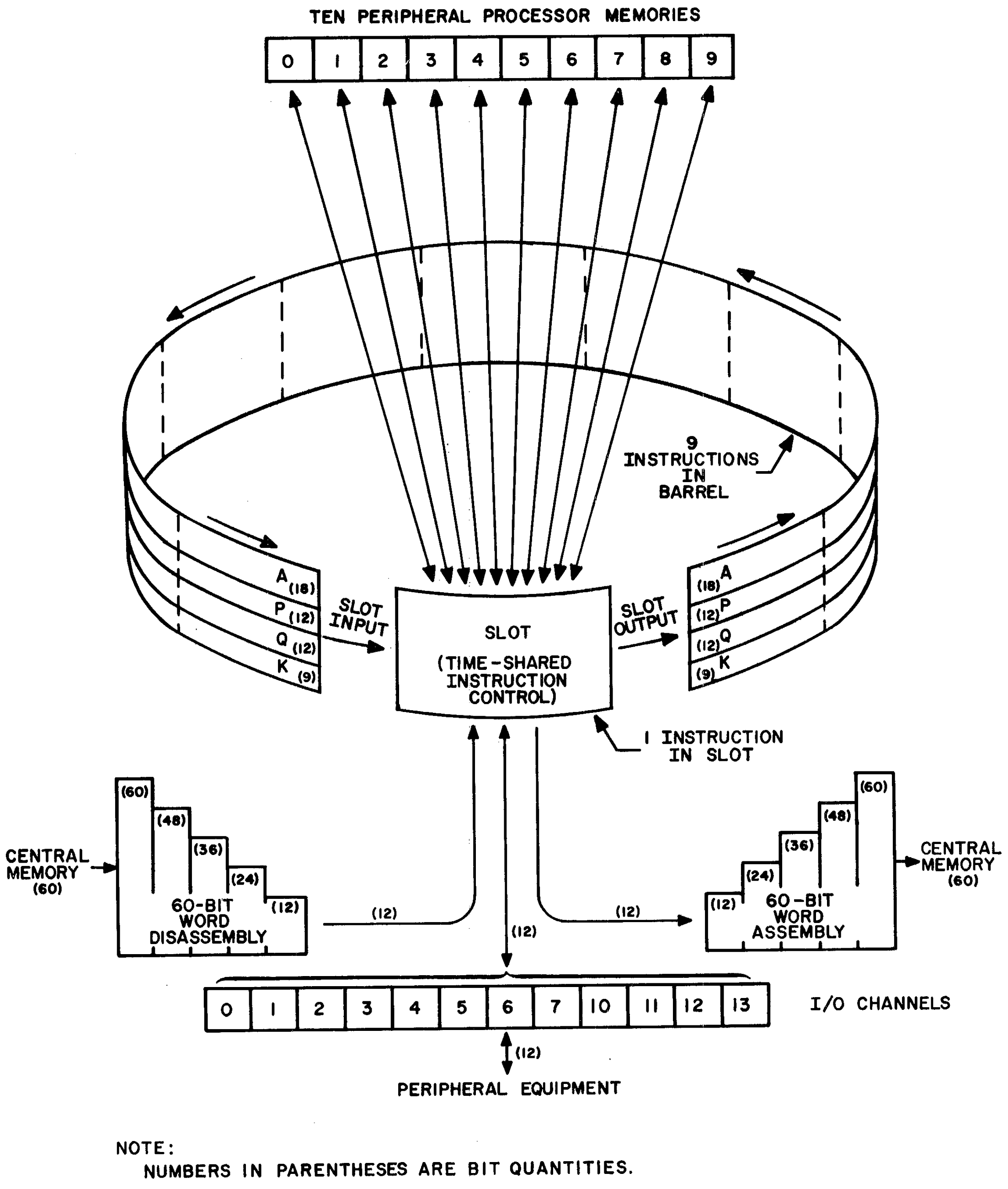


Figure 2-6. Barrel and Slot Operation

A Register

The 18-bit A register holds one operand for arithmetic, logical, or selected I/O operations. The content of the A register may be an arithmetic operand, CM address, I/O function, or I/O data word. Various instructions operate on 6, 12, or 18 bits of the A register.

When the A register is used as the CM address, parity is generated for transmission with the address to CMC.

P Register

The 12-bit P register is the program address register.

Q Register

The 12-bit Q register holds data for several functions such as the address of the operand during direct addressing, address of the operand during indirect addressing, peripheral address of data used during one-word central read or write instructions, upper six bits during constant mode instructions, channel number on all I/O and channel branch instructions, shift count, and relative jump designator.

K Register

The nine-bit K register holds a six-bit f portion of an instruction word and a three-bit trip count. The trip count determines the operation of an instruction at different stages of completion.

INPUT/OUTPUT

Any PP can access any of the 12 bidirectional I/O channels of a PPS or any of the 24 bidirectional channels of an expanded system. All PPs communicate with external equipment and each other through the independent I/O channels. Each channel may be connected to one or more pieces of external equipment, but only one piece of equipment can use a channel at one time. All channels can be active simultaneously.

Each I/O channel transfers a 12-bit word plus 1 parity bit at rates up to one word every microsecond when the PPs are operating at 1X speed. When the PPs operate at 2X speed, channel transfers occur at a rate up to one word every 500 nanoseconds with one exception. This exception occurs only in a 14-, 17-, or 20-PP system and is described in Channel Conflicts in an Expanded PP System in section 5.

Pulse communication is used on all data and control lines of a channel. All control lines are synchronized to the PP clock system.

An unanswered I/O or CM request from a PP causes the PP to hang, causing the PP to operate in a loop. The loop makes the PP continually look for a reply, keeping the PP from proceeding to other operations. The PP may be released from the hung condition by a deadstart, or if CM was the cause, by control bit 78 or 79 in the status and control register.

Parity is generated on the output channels and is checked on the input channels. If a parity error is detected on input data transfer, a bit is set in the status and control register. The status and control register channel parity error status bits are not set on output data transfer parity errors. Each channel is provided with a switch to disable checking parity on input data from external devices that have no parity capability.

Data flows between a PPM and the external device in blocks of words. A block may be as small as one word. A single word may be transferred between an external device and a PP A register.

The channel instructions direct all activity with external equipment. These instructions read the status and provide a selection of an external device on any channel and transfer data to and from the selected device. Two channel conditions available to all PPs to aid in orderly use of channels are:

- Each channel has an active/inactive flag to signal that the channel has been selected for use and is busy with an external device or another PP.
- Each channel has a full/empty flag to signal that a word (function or data) is available in the register associated with the channel.

STATUS AND CONTROL REGISTER

The status and control register is part of PPS-0. A second abbreviated status and control register is part of the optional PPS-1.

Status bits are externally set. They are tested, cleared, and logged by software. Status register bits can be read in 12-bit words using function codes and an input instruction. Control register bits can be set and tested individually or read as 12-bit words. Section 5 contains additional information, including Table 5-9 which lists the status and control register bit assignments.

The following information describes each bit or group of bits in the status and control register and should be used with Table 5-9.

Read Pyramid Parity Error (Bit 0) — Status

This bit indicates a parity error on data transmitted from CMC to PPS. The bit also indicates a CM parity error during parity mode operation.

CSU-O Address Parity Error (Bit 1) — Status

This bit indicates a parity error on the address transmitted from CMC to CSU-0.

CSU-1 Address Parity Error (Bit 2) — Status

This bit indicates a parity error on the address transmitted from CMC to CSU-1.

SECDED Error (Bit 3) — Status

This bit indicates that a single-error correction or a double-error detection has occurred. The bit also locks bits 40 through 53. These bits are unlocked, but not reset, when bit 3 is reset (cleared). Bit 3 must be reset by software to detect further SECDED status. Bit 183 identifies the SECDED error as a single error, when cleared, or a double error, when set. Bit 118, if set, inhibits bit 3 for single errors but not for double errors.

(Bit 4 Not Used)

CMC Parity Error (Bit 5) — Status

This bit indicates that an address or data transmission parity error has been received by CMC. The bit is used in conjunction with bits 54, 55, and 139. The bit locks bits 54, 55, and 139 so that their status cannot be modified until bit 5 is cleared. Bit 5 must be reset by software to detect further CMC parity errors.

Parity Error on Data Received from External Channel (Bit 6) — Status

This bit indicates that a PP in one PPS chassis detected a parity error that was received from a channel in the second PPS chassis in an expanded system. A PP in the second PPS chassis may detect a parity error that was received from a channel in the first PPS chassis. Therefore, separate inter-PPS parity errors may occur, one on channel 16 and one on channel 36.

Parity Error on Data Transmitted from External PP (Bit 7) — Status

This bit indicates that a PP in one PPS chassis detected a parity error on data transmitted from a PP in the second PPS chassis in an expanded system. A PP in the second PPS chassis may detect a parity error on data transmitted from a PP in the first PPS chassis. Therefore, separate inter-PPS parity errors may occur, one on channel 16 and one on channel 36.

CSU-0 Fault and CSU-1 Fault (Bits 8, 9) — Status

These bits indicate that a CSU has detected a CM timing error. The CSU detects irregularities in CM refreshes that may occur too soon or last too long. Either condition causes excessive power dissipation. The CSU is disabled until it receives a master clear signal.

Error in Second PPS (Bit 10) — Status

This bit indicates that one or more of bits 0 through 39 in the PPS-1 status and control register is set.

ECS Error (Bit 11) — Status

This bit indicates that an error occurred on an ECS transfer. The type of error is indicated by the status locked in bits 136, 137, and 138 by bit 11. Bit 11 must be reset to detect further errors.

CP-0 P Register Parity Error (Bit 12) — Status

This bit indicates that the PPS detected a parity error on a read of the P register for CP-0.

CP-1 P Register Parity Error (Bit 13) — Status

This bit applies only to model 174 and is unused in models 172, 173, and 175. The bit indicates that the PPS detected a parity error on a read of the P register for CP-1.

PP0 through PP9 Memory Parity Error (Bits 14 through 23) — Status

These bits indicate the occurrence of a PPM parity error in a PP. The bits indicate, on a one-to-one basis, the status of each PP as shown in Table 5-9. The bits indicate the logical PP numbers and are not affected by a reconfiguration of the PPM because of resetting the PPS-1/PPS-0 and PPS SELECT switches on the deadstart panel.

Channel 0 through 13 Parity Error (Bits 24 through 35) — Status

These bits indicate the occurrence of a parity error in the corresponding I/O channel. Each bit indicates the status of one channel as shown in Table 5-9.

Mains Power Failure (Bit 36) — Status

This bit indicates that the primary power mains feeding the computer system are de-energized and have remained so for more than one-half cycle (16.6 milliseconds for 60-Hz power and 10.0 milliseconds for 50-Hz power) of the mains frequency. If power returns, the line feeding the bit automatically goes false, within one cycle of the mains frequency. The bit must be cleared by software.

Shutdown Imminent (Bit 37) — Status

This bit indicates one of the following conditions.

1. Primary power mains feeding the system are deenergized and have remained so for at least 100 milliseconds. Power probably will not return to normal within the regulation range of the system secondary power supply, normally a motor-generator set.
2. An environmental condition (including dew point warning and chassis temperature) is abnormal and approaching an emergency power shutdown.
3. An environmental condition is changing at an abnormally high rate.
4. An environmental condition is about to execute a controlled power shutdown.
5. A critical system device is down because of environmental conditions. (This indication exists only if the system has monitoring provisions for the device.)

If power and environmental conditions return to normal, except in the case of an emergency shutdown limit, the line feeding the bit automatically goes false within one cycle of the mains frequency. The bit must be cleared by software.

When both the mains power failure and power shutdown imminent bits are set, one of the following coincident conditions exists.

1. A power mains failure has occurred for longer than approximately 100 milliseconds. Power will probably not return within the regulation range of the system secondary power supplies. The kernel system (CP, all PPs, all channels, store, all first level controllers, and all system disk units) remains available for processing for the balance of the motor-generator ride-through after the shutdown imminent bit sets. In this case, the mains power failure bit sets at least 50 milliseconds before the shutdown imminent bit sets. However, all peripheral equipment powered directly from the mains has probably failed.
2. A controlled shutdown limit has been reached. The limit sensor has disconnected the primary power mains from the system secondary power supply, and the kernel system processing remains available for the balance of the motor-generator ride-through after the shutdown imminent bit sets. In this case, the mains power failure and the shutdown imminent bits set at approximately the same time.

Examples of possible conditions are:

	Condition	Explanation
1.	Mains power failure only; power returns.	Indicates that all peripheral equipment powered directly from the mains has probably failed. The system is not down, but user intervention is necessary to restore power to affected peripherals.
2.	Mains power failure and shutdown imminent.	Indicates the system will probably terminate and require restart.
3.	Mains power failure and shutdown imminent, mains power failure bit clears, or the mains power failure and shutdown imminent bits clear.	The explanation for condition 1 applies. This is a rare occurrence and may not be a stable condition.
4.	Shutdown imminent, no mains power failure.	Either a shutdown timeout (1 to 2 minutes) is in progress because of an environmental problem, or a warning level has been reached which ultimately requires user intervention. Sufficient time may exist for the user or software, if software provides, to initiate and complete system checkpoints. If the mains power failure bit 36 sets later, a timeout has been completed and the system behaves as though an emergency shutdown limit was reached.

(Bits 38 and 39 Not Used)

Syndrome Bits 0 through 7 (Bits 40 through 47) — Status

These bits, along with bits 48 through 53, are provided by CMC upon detection of a SECDED error. They provide the information needed to isolate a single-error failure to a particular array module. Bits 40 through 47 are locked by the setting of bit 3. Clearing bit 3 unlocks bits 40 through 47 but does not clear them.

Syndrome Bits 0 through 5 (Bits 48 through 53) — Status

These bits, along with bits 40 through 47, are provided by CMC upon detection of a SECDED error. Bits 48, 49, and 50 indicate the CM bank number. Bits 51 and 52 indicate the CM quadrant. Bit 53 indicates the CSU chassis number. Bits 48 through 53 are locked by the setting of bit 3. Clearing bit 3 unlocks bits 48 through 53 but does not clear them.

Parity Error Port Code Bits 0, 1 (Bits 54, 55) — Status

These bits indicate which CMC port had a parity error. The bits are locked by the setting of bit 5 and cannot be modified until bit 5 is cleared.

Bit 55	Bit 54	Port
0	0	CP-1 (model 174 only)
0	1	CP-0
1	0	PPS-1
1	1	PPS-0

Breakpoint Port Code Bits 0 , 1 (Bits 56 , 57) — Status

These bits indicate the CMC port that satisfied the breakpoint condition. The bits are locked by the setting of bit 77 and cannot be cleared until bit 77 is cleared.

Bit 57	Bit 56	Port
0	0	CP-1 (model 174)
0	1	CP-0
1	0	PPS-1
1	1	PPS-0

Breakpoint Function Code Bits 0,1 (Bits 58,59) — Status

These bits indicate what type of instruction caused the breakpoint condition to be satisfied. The bits are locked by the setting of bit 77 and cannot be modified until bit 77 is cleared.

Bit 59	Bit 58	Type
0	0	Read
0	1	Write
1	0	RNI
1	1	This condition does not occur.

PPS P Register Bits 0 through 11 (Bits 60 through 71) — Status

These bits indicate the content of the P register for the PP that satisfied the breakpoint condition when bits 76 and 83 are set. When bit 83 is not set, the bits display the P register of the selected PP. The PP selection can be made manually by switches or through software selection of control bits 120 through 124. Bits 60 through 71 are locked by the setting of bit 76 and cannot be modified until bit 76 is cleared.

PP Code Bits 0 through 3 (Bits 72 through 75) — Status

These bits indicate which PP stored the content of its P register in bit positions 60 through 71. Bits 72 through 75 are locked by the setting of bit 76 and cannot be modified until bit 76 is cleared. Bit 83 is associated with bits 72 through 75.

PPS Breakpoint Bit (Bit 76) — Status

This bit, with bit 83 set, indicates that the breakpoint address was referenced by PPS. The P register content of the referencing PP is locked into bit positions 60 through 71. Also, the referencing PP code is locked into bit positions 72 through 75. These bits are locked so that their status cannot be modified until bit 76 is cleared. Bit 76 must be reset by software to detect further PPS breakpoint addresses. With bit 83 clear, the content of the P register of the PP selected by bits 120 through 124 is made available for monitoring by bits 60 through 71. The P register status is not locked but continually tracks the program address of the selected PP.

CMC Breakpoint Match (Bit 77) — Status

This bit indicates that the breakpoint condition occurred. The breakpoint condition is defined by the absolute address (located in bits 96 through 113) and the breakpoint condition code (located in bits 114 through 117). It also locks bits 56 through 59 so that their status cannot be modified until bit 77 is cleared. Bit 77 must be reset by software to detect further breakpoint conditions.

Clear Central Memory Busy (Bit 78) — Control

This bit clears CM busy and unhangs a PP on an unanswered CM request. The bit causes a one-shot operation. The bit must be cleared by software and set again to execute its function a second time.

Set C5 Full (Bit 79) — Control

This bit acts as a C5 (input read register) full signal and unhangs a PP on an unanswered CM read request. The full signal simulates a C5 full condition, making the hung PP appear to get a response. Data read during the response is undefined. The bit causes a one-shot operation. The bit must be cleared by software and set again to execute its function a second time.

Force Zero Parity on Channels (Bit 80) — Control

This bit forces the data parity bits in the I/O channels to zero. A deadstart clears the bit.

Force Zero Parity on PP Memories (Bit 81) — Control

This bit forces the PPM parity bits to zero. A deadstart clears the bit.

(Bit 82 Not Used)

PPS Breakpoint Mode Select (Bit 83) — Control

This bit, when set, forces the P register field (bits 60 through 75) into breakpoint mode. When clear, it forces the P register field into program address display mode. The breakpoint field is locked by the setting of bit 76. A deadstart clears the bit.

All PPs 500 - Nanosecond Major Cycle (Bit 84) — Control

This bit selects the major cycle time for all PPS in PPS-0 and PPS-1. When set, the cycle time is 500 nanoseconds. When clear, the cycle time is 1000 nanoseconds. A deadstart clears the bit.

Inhibit PPS Request to CMC (Bit 85) — Control

This bit prevents any PP from making a read/write/exchange request during the time the CMC clears, ensuring that the master clear does not hang any PPS. A deadstart clears the bit.

(Bits 86 through 94 Not Used)

Stop on PPM Parity Error (Bit 95) — Control

This bit, when set, causes a PP to stop if it detects a parity error in its PPM. The affected PP goes into a read mode, constantly reading the address in which the error occurred. If a second bit failure occurs, the affected PP continues in the read mode. The associated PPM error status also sets when the error initially occurs. After being tested and cleared by software, the error status does not set again while the PP is stopped. A deadstart clears the bit.

Breakpoint Address Bits 0 through 17 (Bits 96 through 113) — Control

These bits define the absolute address to be used for the breakpoint condition, defined by bits 114 through 117. Bits 96 through 113 are sent to CMC and compared with all addresses being accessed.

Breakpoint Condition Code Bits 18 through 21 (Bits 114 through 117) — Control

These bits define the breakpoint conditions.

Bit 117	Bit 116	Bit 115	Bit 114	Condition
X	X	0	0	Read
X	X	0	1	Write
X	X	1	0	RNI
X	X	1	1	Any of the above
0	0	X	X	Disabled
0	1	X	X	Enabled for PPS
1	0	X	X	Enabled for CP
1	1	X	X	Enabled for PPS or CP

Inhibit Single—Error Report (Bit 118) — Control

This bit, when set, stops the recording of single-error status information and blocks setting bit 3 of the status and control register if a single error occurs. Double errors continue to set bit 3 and be reported by bit 183.

(Bit 119 Not Used)

PP Select Code Bits 0 through 3 (Bits 120 through 123) — Control

These bits determine which PP is forced to exit (bit 125), deadstart (bit 126), or display (when bit 83 is clear) its P register. A deadstart clears bits 120 through 123.

PP Select Auto/Manual Mode (Bit 124) — Control

This bit selects the mode of selection. When set, PP selection is under program control. PP selection is then made by bits 120 through 123. When clear, selection is manual, and the PP selection is made by switches on the PP chassis at location J40. A deadstart clears the bit.

Force Exit on Selected PP (Bit 125) — Control

This bit clears a selected hung PP by forcing an instruction exit. The PP resumes operation at its next slot time at P plus 1. A forced instruction exit occurs once each time bit 125 sets. The bit causes a one-shot operation. The bit must be cleared by software and set again to cause a second exit. A deadstart clears the bit.

Force PP Deadstart on Selected PP (Bit 126) — Control

This bit, along with control bits 120 through 124, provides a programmable capability to make individual PP deadstarts. Bits 120 through 124 select the PP, and bit 126 forces the selected PP into a deadstart input condition. The selected PP then goes through the same deadstart sequence as would occur under a hardware-controlled deadstart. The PP is set up for a 71XX instruction, where XX is the selected PP number. This instruction causes the PP to attempt an input on its own channel. The software must first ensure that the selected channel is empty and active at the time of the deadstart. No other I/O operation can be in process on the channel. The master clear signal to the channel is inhibited. The selected PP remains in the deadstart condition until bit 126 is cleared. A system deadstart clears bit 126.

CSU, CMC, CP Master Clear (Bit 127) — Control

This bit, when set, sends a master clear to the CSU, CMC, and CP chassis (two CP chassis for model 174). The bit is OR'd with the deadstart signal. The bit or the deadstart signal causes a master clear. The bit holds the CMC and CP chassis in a cleared state as long as it is set. The PP chassis is not affected by this bit, unless a PP is making a CM reference. To avoid hanging any PP, bit 85 should be set before bit 127. A deadstart clears the bit.

Force Zero SECDED Code and Parity CMC to CM (Bit 128) — Control

This bit forces the CMC to put a zero check code and parity bit on data being sent to CM. It also forces CMC to put a zero parity bit on data transmitted to a requesting unit such as ECS.

Force Zero Address Parity CMC to CM (Bit 129) — Control

This bit forces CMC to put a zero parity bit on the address being sent to CM.

Disable Address Parity Error (Bit 130) — Control

This bit disables address parity error detection at the CSU. This prevents a condition where reads or writes are inhibited during the presence of any address parity error.

(Bit 131 Not Used)

Force Zero Parity Code 0 (Bit 132) and Code 1 (Bit 133) — Control

These bits force a zero parity bit on the following transmission paths.

Bit 133	Bit 132	Transmission Path
0	0	Normal parity
0	1	Address from ECS to CMC
1	0	Word count or address from CP to ECS coupler
1	1	Data from ECS to CMC

Refresh Margin Slow (Bit 134) — Control

This bit, when set, decreases the normal CM refresh rate from once every 25.6 microseconds to once every 32 microseconds.

Refresh Margin Fast (Bit 135) — Control

This bit, when set, increases the normal CM refresh rate from once every 25.6 microseconds to once every 19.2 microseconds.

ECS Transfer Error Codes 0 through 2 (Bits 136 through 138) — Status

These bits indicate errors that occur during an ECS transfer. The following list gives the status-bit code that states where the error occurred. The bits are locked by the setting of bit 11.

Bit 138	Bit 137	Bit 136	Status
0	0	0	Not used
0	0	1	CMC data parity error
0	1	0	CMC double error
0	1	1	CMC to CM address parity error
1	0	0	CP to ECS coupler parity error
1	0	1	ECS bank parity error
1	1	0	ECS controller data parity error
1	1	1	ECS controller address parity error (This indicates no error when bit 11 is clear.)

CMC Address/Data Parity Error (Bit 139) — Status

This bit indicates an address parity error in CMC. The bit is used with bits 5, 54, and 55. If the bit is clear and bit 5 is set, the CMC parity error is a data error. Bit 139 is locked by the setting of bit 5 and cannot be modified until bit 5 is cleared.

(Bit 140 Not Used)

Clock Frequency Magnitude 0,1, and Slow/Fast (Bits 141 through 143) — Control

These bits are for use in maintenance operations. The bits form a 3-bit code that sets the frequency margins of the basic 40-MHz clock. A 20-MHz clock and a 10-MHz clock originate from the basic clock and change frequency margins by the same percentage as the basic clock. A deadstart clears these bits.

The 3-bit code translations and the resulting margins for each of the clocks are as follows:

Bit 143	Bit 142	Bit 141	40-MHz Clock	20-MHz Clock	10-MHz Clock
0	0	0	40.000	20.000	10.000
0	0	1	39.375	19.688	9.844
0	1	0	38.750	19.375	9.688
0	1	1	38.125	19.063	9.531
1	0	0	40.000	20.000	10.000
1	0	1	40.625	20.313	10.156
1	1	0	41.250	20.625	10.313
1	1	1	41.875	20.938	10.469

Reference Voltage Margin Address Bit Status (Bits 144 through 149) — Status

These bits apply only to model 175 and are unused in models 172 through 174. The bits indicate which CP chassis quadrant address is selected for a reference voltage margin (RVM). The bits verify operation of reference margin addressing and correspond one-to-one with control bits 168 through 173.

Reference Voltage Margin Hi/Lo and All/One (Bits 150,151) — Status

These bits apply only to model 175 and are unused in models 172 through 174. Bit 150 indicates that the RVM is low when clear and high when set. Bit 151 indicates that one CP module is selected for RVM when clear and that all CP modules are selected for RVM when set. The bits verify operation of the reference margin selections and correspond with bits 154 and 155, respectively.

Clock Pulse Width Narrow and Wide (Bits 152,153) — Control

These bits apply only to model 175 and are unused in models 172 through 174. The bits control the clock pulse width according to the following bit translations.

Bit 153	Bit 152	Clock Pulse
0	0	Normal
0	1	Narrow
1	0	Wide
1	1	Narrow

The 152 and 153 bit outputs are in parallel with the CLOCK PULSE switch on CP chassis 5. The CLOCK PULSE switch must be in the normal position (middle) to permit clock pulse margin control from the status and control register. In the narrow or wide position, the CLOCK PULSE switch overrides the status and control register clock pulse width bits.

Select Hi/Lo Reference Voltage Margins (Bit 154) — Control

This bit applies only to model 175 and is unused in models 172 through 174. When set, the bit selects the high RVM for the CP modules selected by bits 155 through 173. When clear, the bit selects the low RVM for the selected modules. If bits 156 through 167 do not reference a CP chassis quadrant (Figure 2-7), bit 154 has no effect.

Select All/One Reference Voltage Margins (Bit 155) — Control

This bit applies only to model 175 and is unused in models 172 through 174. When this bit is set and bits 168 through 173 are set, the RVM for all CP modules within the quadrants selected by bits 156 through 167 are simultaneously selected. When clear, bit 155 permits RVM to be applied to individual modules within the quadrants selected by bits 156 through 173. If bits 156 through 167 do not reference a CP chassis quadrant (Figure 2-7), bit 155 has no effect.

Reference Voltage Margins Quadrant Select 0 through 11 (Bits 156 through 167) — Control

These bits apply only to model 175 and are unused in models 172 through 174. The bits determine, on a one-to-one basis, which quadrant or quadrants (Figure 2-7) of a CP chassis receive an RVM. For example, select 3 selects quadrant 3, and select 8 selects quadrant 8. Bits 156 through 167 are associated with bits 154, 155, and 168 through 173.

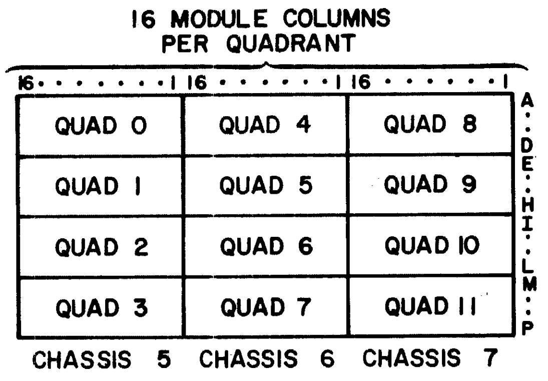


Figure 2-7. Model 175 CP Chassis Quadrants (Viewed from Module Side)

Reference Voltage Margins Module Address Bits 0 through 5 (Bits 168 through 173) — Control

These bits apply only to model 175 and are unused in models 172 through 174. The bits select one of 64 modules in a CP chassis quadrant (Figure 2-7). Address bits 0 through 3 select 1 of 16 module columns. Address bits 4 and 5 select 1 of 4 module rows. The addresses increase by module location within a row and by rows within a quadrant.

PPS to CMC Zero Address Parity (Bit 174) — Control

This bit forces the PPS to put a zero parity bit on the address sent to CMC.

PPS to CMC Zero Data Parity (Bit 175) — Control

This bit forces the PPS to put a zero parity bit on the data sent to CMC.

(Bits 176 through 182 Not Used)

Double Error (Bit 103) — Status

This bit, when set, indicates that a double error occurred. When the bit is clear and bit 3 is set, it indicates that a single error set bit 3. When a SECDED error occurs, one of the following conditions describes the error.

- A single-bit error occurred.

  Bit 3 is set, indicating a SECDED error.

  Bit 183 is clear, indicating a single error.

  Bits 40 through 47 contain the syndrome code (odd number of bits) indicating the failing bit.

  Bits 48 through 53 indicate the failing CM bank, quadrant, and chassis.

- A double-bit error occurred.

  Bit 3 is set, indicating a SECDED error.

  Bit 183 is set, indicating a double error.

  Bits 40 through 47 contain a syndrome code (even number of bits) that does not indicate the failing bits.

  Bits 48 through 53 indicate the failing CM bank, quadrant, and chassis.

- A single-bit error occurred. Before software could clear it, a double-bit error occurred.

  Bit 3 is set, indicating a SECDED error.

  Bit 183 is set, indicating a double error.

  Bits 40 through 47 contain a syndrome code (odd number of bits) for the single-bit error.

  Bits 48 through 53 indicate the failing CM bank, quadrant, and chassis for the single error.

CP-0 to CMC Zero Address Parity (Bit 184) — Control

This bit applies only to models 172 through 174 and is unused in model 175. The bit forces the CP to put a zero parity bit on the address sent to CMC.

CP-1 to CMC Zero Address Parity (Bit 185) — Control

This bit applies only to model 174 and is unused in models 172, 173, and 175. The bit forces CP-1 to put a zero parity bit on the address sent to CMC.

CP-0 to CMC Zero Data Parity (Bit 186) — Control

This bit applies only to models 172 through 174 and is unused in model 175. The bit forces CP-0 to put a zero parity bit on the data sent to CMC.

CP-1 to CMC Zero Data Parity (Bit 187) — Control

This bit applies only to model 174 and is unused in models 172, 173, and 175. The bit forces CP-1 to put a zero parity bit on the data sent to CMC.

Software Flag 0 and Flag 1 (Bits 188,189) — Control

These bits are used by diagnostic software for communication between PPs.

(Bits 190 and 191 Not Used)

CP-0 Stopped (Bit 192) — Status

This bit, when set, indicates that the CP has stopped. When the CP resumes operation, the bit clears.

CP-1 Stopped (Bit 193) — Status

This bit applies only to model 174 and is unused in models 172, 173, and 175. When set, the bit indicates that the CP-1 has stopped. When the CP resumes operation, the bit clears.

ECS in Progress Flag (Bit 194) — Status

This bit indicates ECS transfer is currently in progress. When the transfer has completed or terminated, the bit clears.

Monitor Flag CP-0 (Bit 195) — Status

This bit indicates the condition of the monitor flag in CP-0.

Monitor Flag CP-1 (Bit 196) — Status

This bit applies only to model 174 and is unused in models 172, 173, and 175. The bit indicates the condition of the monitor flag in CP-1.

PPM Reconfiguration Bits 0 through 4 (Bits 197 through 201) — Status

These bits indicate the positions of the PPS-1/PPS-0 and PPS SELECT switches on the deadstart panel. The switches select which physical PPM is logical PPM-0. The PP associated with the selected PPM is the controlling PP-0. The status and control register bit 201 indicates that PPS-0 is selected when the bit is 0 and that PPS-1 is selected when the bit is 1. A PPM reconfiguration is not effective in PPS-1 unless all 10 PPs are installed. Bits 197 through 200 indicate the PP selection as follows:

Bit 200	Bit 199	Bit 198	Bit 197	Selection
0	0	0	0	PP-0
0	0	0	1	PP-1
0	0	1	0	PP-2
0	0	1	1	PP-3
0	1	0	0	PP-4
0	1	0	1	PP-5
0	1	1	0	PP-6
0	1	1	1	PP-7
1	0	0	0	PP-8
1	0	0	1	PP-9

(Bits 202 and 203 Not Used)

VISUAL STATUS/ERROR INDICATORS

Some status bits and some control bits in the status and control register are displayed by special light modules. The light modules and the bits they display are described in PPS Light Modules in section 3.

REAL-TIME CLOCK

The computing system contains a real-time clock. The clock may be used to determine program running time, as a reference to track the time-of-day, or for other functions determined by the computer programs.

The clock runs continuously during computer power application. Output from the clock comes from a 12-bit register that increments once every microsecond to the maximum capacity of the register (4096 microseconds). When the register reaches capacity, it resets and continues counting. The counting cannot be preset or altered.

Any of the PPs may read the 12-bit clock output with the input to A channel d, 70 instruction. The instruction permits access to the clock on internal channel 14 (octal). Any attempts to output information on channel 14 cause it to hang.

DATA CHANNEL CONVERTERS

Each system DCC attaches to a data channel of the PPS (Figure 2-8). A DCC may share the data channel with up to seven other pieces of CDC 6000/CYBER Series peripheral equipment. Up to eight 3000 Series controllers can be attached to one DCC.

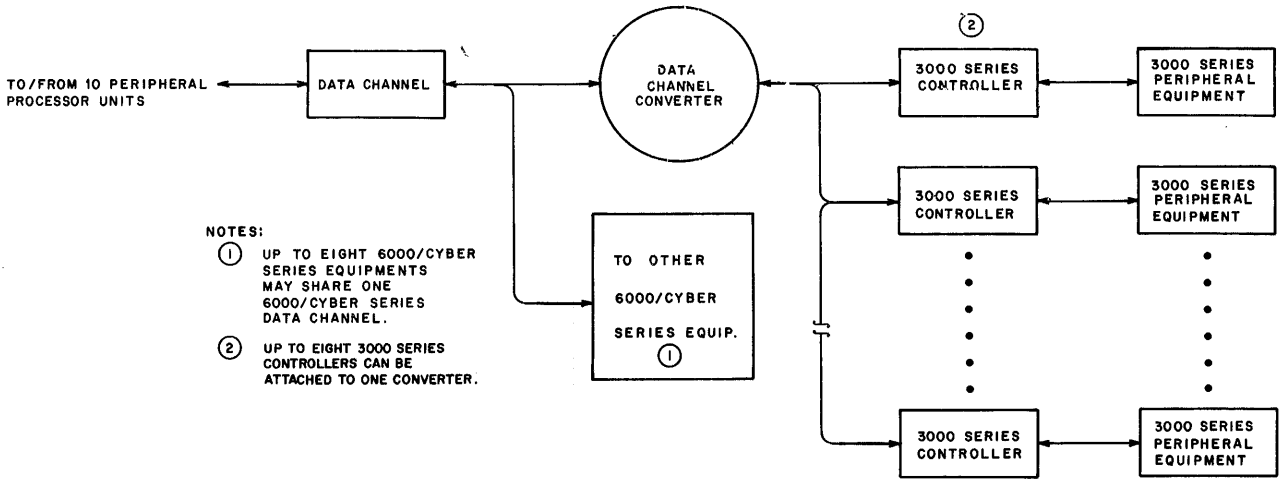

TO/FROM 10 PERIPHERAL PROCESSOR UNITS
DATA CHANNEL
DATA CHANNEL CONVERTER
TO OTHER 6000/CYBER SERIES EQUIP. ①
② 3000 SERIES CONTROLLER
3000 SERIES CONTROLLER
3000 SERIES CONTROLLER
3000 SERIES PERIPHERAL EQUIPMENT
3000 SERIES PERIPHERAL EQUIPMENT
3000 SERIES PERIPHERAL EQUIPMENT
NOTES:
① UP TO EIGHT 6000/CYBER SERIES EQUIPMENTS MAY SHARE ONE 6000/CYBER SERIES DATA CHANNEL.
② UP TO EIGHT 3000 SERIES CONTROLLERS CAN BE ATTACHED TO ONE CONVERTER.

Figure 2-8. Data Channel Converter Configuration

To prepare any of the 3000 Series equipment for operation, the DCC must first be selected. Then the desired 3000 Series equipment is selected (connected). The two select operations are made by function codes sent from a PPU through the data channel. A data channel is part of the I/O channel that exists between a PP and external equipment which uses the same type transmitters and receivers for information interchange. The DCC differs from other CDC 6000/CYBER Series equipment as follows:

- The DCC must be attached to the data channel ahead of all other CDC 6000/ CYBER Series devices.
- The DCC does not replay (pass on) information to other equipment on the same data channel when selected. This prevents unwanted activity in the other equipment caused by identical function codes.
- The DCC must be deselected (2100) before other CDC 6000/CYBER Series equipment sharing the data channel can be selected.
- A master clear (MC) signal on deadstart operations selects all DCCs in the computer system.

3000 SERIES INTERRUPT FEATURE

All 3000 Series peripheral equipment has an interrupt feature which enables them to notify the DCC when specific operating conditions occur. Most of the peripherals use interrupt conditions which are selected or released in an equipment by function codes which are:

- Interrupt on ready, or interrupt on ready and not busy
- Interrupt on end of operation
- Interrupt on abnormal end of operation

The reference manual describing each 3000 Series equipment provides the interrupt select function codes and defines the interrupt conditions.

The 3000 Series equipment sends an interrupt signal to the DCC and sets a corresponding bit in the DCC status word when one of the selected interrupt conditions occurs. Bits 3 through 10 of the 12-bit status word indicate interrupts from any one of the eight possible pieces of equipment served by the DCC. The status bit set depends upon the equipment number of the device sending the interrupt.

Equipment Number	DCC Status Bit
0	3
1	4
2	5
3	6
4	7
5	8
6	9
7	10

Peripheral equipment need not be connected to the DCC to send an interrupt signal to it. Thus, the interrupt feature provides a limited status check for an equipment even though it is not connected.

An interrupt status bit in the DCC is present (set) as long as the equipment maintains the interrupt signal. An interrupt signal is cleared by any one of the following.

- A DCC master clear (MC) function (1700). This clears all 3000 Series equipment attached to the DCC and the DCC itself.
- A function code sent to the interrupting equipment.
- A deadstart MC signal from the CDC 6000/CYBER data channel.

BUFFER FLUSHING

The buffer-flush feature allows the DCC to terminate the PP I/O buffer when an interrupt on abnormal end of operation condition exists in the peripheral equipment. To enable this, the peripheral equipment must be set to interrupt on an abnormal end of operation. This action sends an interrupt override signal to the DCC. The interrupt override signal initiates the buffer-flush operation by forcing full or empty signals to the PP until the I/O buffer is terminated.

DISPLAY CONTROLLER

The display controller provides the display station with analog and digital signals that direct the writing of symbols on the display station cathode-ray tube (CRT). The controller provides analog-symbol signals and digital-position, unblank, and size-selection signals.

The analog-symbol signals cause small-scale deflection of the CRT beam for tracing symbols on the face of the CRT. Four lines carry the signals to the display station. Two of the lines are for the x (horizontal) deflection and two are for the y (vertical) deflection.

The digital-position signals cause large-scale deflection of the CRT beam for positioning the symbols on the face of the CRT. The signal lines to the display station carry nine bits for the beam x deflection and nine bits for the beam y deflection.

The unblank signal enables the CRT beam only during the time an analog-symbol signal is causing a symbol trace. The unblank signal is a pulse train that is synchronized with the symbol signal.

The size-selection signal is binary-coded. It is carried on two lines and provides the selection of one of three symbol sizes.

Eight other lines between the display controller and display station carry control signals and display station keyboard character codes.

OPERATING INSTRUCTIONS 3

This section describes the mainframe controls and indicators and the operating procedures which are hardware-dependent. Software-dependent procedures are in system software reference manuals.

The section does not include power-on and power-off procedures. Operators are generally restricted from using these procedures, except for the use of the system EMERGENCY OFF switch in an emergency.

CONTROLS AND INDICATORS

Control and indicator descriptions are grouped by their locations within the mainframe. These locations are:

- Deadstart panel
- Central processor (CP) chassis, model 175
- Central memory control (CMC) chassis, models 172 through 174
- Central memory control (CMC) chassis, model 175
- Extended core storage (ECS) coupler chassis, model 175
- Central storage unit (CSU) chassis
- Input/output (I/O) connector panel
- Peripheral processor subsystem (PPS) chassis

DEADSTART PANEL

The deadstart panel (Figure 3-1) is located in bay 1 to the right of chassis 1, as seen when facing chassis 1. The panel contains PPS control switches which are only active during a deadstart. The switches and their functions are listed in Table 3-1.

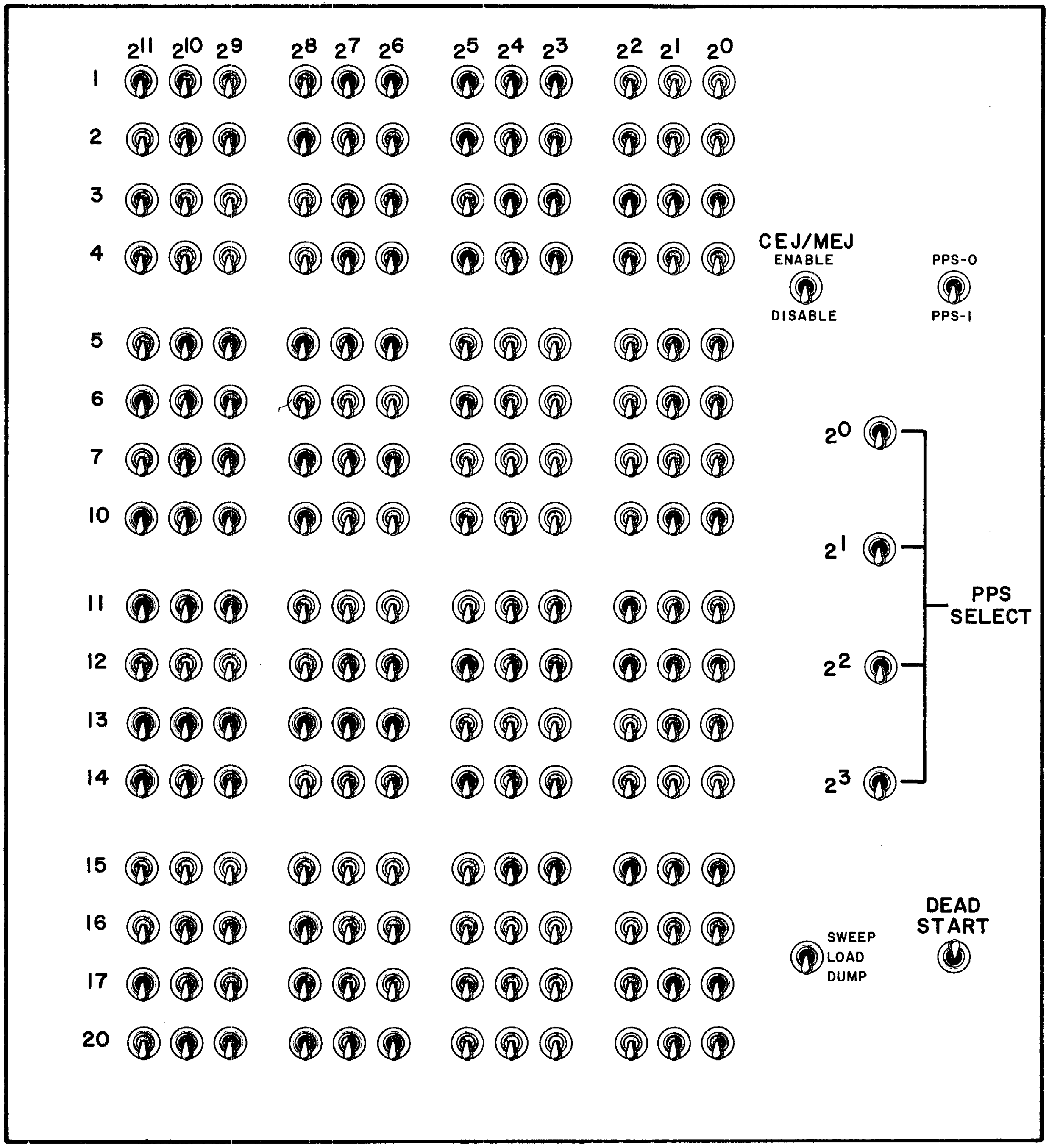


Figure 3-1. Deadstart Panel

TABLE 3-1. DEADSTART PANEL CONTROLS

Panel Nomenclature	Description	Function
2^0 through 2^{11} by 1 through 20	Toggle switch matrix	Provides a 16-word deadstart program for PP-0. Switches 2^0 through 2^{11} set 12 bits for each of the program words, labeled 1 through 20 (octal). Up position sets bit. Down position clears bit.
CEJ/MEJ ENABLE DISABLE	Toggle switch	Enables or disables the central exchange jump (CEJ) instructions for the CP and the monitor exchange jump (MEJ) instructions for the peripheral processors (PPs). The switch position is set prior to a deadstart. Any resetting of the switch after a deadstart does not affect the computer operation until the next deadstart.
PPS-0 PPS-1	Toggle switch	Selects PPS-0 or PPS-1 as the subsystem that will contain the controlling PP-0.
PPS SELECT 2^3, 2^2, 2^1, 2^0	Toggle switches	Permit the selection of any peripheral processor memory (PPM) as PPM-0. The PP corresponding to the selected PPM becomes the control unit PP-0. If PP-0 malfunctions, the user may use the switches to select any of the other nine PPs as PP-0. The selection renumbers all of the PPs, with the selected PP being PP-0. The selection retains the PP order of rotation in the barrel and slot matrix. (Refer to Barrel and Slot in section 2.) The selection is made by enabling the switches to form a binary number of the PP chosen to be PP-0 (for example, 0101 selects PP-5).

TABLE 3-1. DEADSTART PANEL CONTROLS (Cont'd)

Panel Nomenclature	Description	Function
		These switches do not affect the PPS-1 chassis unless that chassis contains all 10 PPs. Up position sets bit. Down position clears bit.
SWEEP LOAD DUMP	Toggle switch	Selects PP-0 mode of operation (refer to Deadstart in this section).
DEAD START	Toggle switch	Provides system deadstart. Up position causes deadstart to repeat each 4096 microseconds which includes a master-clear duration of 1.0 microsecond. Down position is deadstart off.

CENTRAL PROCESSOR CHASSIS MODEL 175

The model 175 CP contains switches and indicators at module locations 5A1 through 5A3 (Figure 3-2). Table 3-2 lists the switches, indicators, and their functions. Table 3-3 provides information for setting the switches for normal or reconfigured CM quadrant configurations.

All CM quadrants, for a particular CM size, are available for use by the model 175 system when the CM configuration switches are set to the normal operation positions shown in Table 3-3.

If one of the 16-bank CM quadrants becomes defective, the CM may be reconfigured to operate without the quadrant. The reconfiguration is performed by determining the defective quadrant and then setting the CM configuration switches for that quadrant to the reconfigured operation switch positions shown in Table 3-3. Any one quadrant may be reconfigured at one time, except quadrant 0 of the 65K and 98K memories. The switches accomplish a logical reconfiguration by manipulating the CM upper address bits.

The switches are always active.

TABLE 3-2. MODEL 175 CP CONTROLS AND INDICATORS

Panel Nomenclature	Description	Function
MEMORY CONFIG S0, S1, S2, S3	Toggle switches	Control CM quadrant configuration. Up positions set bits. Down positions clear bits.
CLOCK PULSE	Toggle switch	Controls clock pulse width. Up position provides wide pulse. Middle position enables software control of pulse width. Down position provides narrow pulse.
WIDE, NARROW	Indicators	Light to show respective clock pulse widths.
ERROR EXCH	Toggle switch	Up position disables CEJ on error exit. Down position enables CEJ on error exit.
DISABLE	Indicator	Lights to show CEJ on error exit condition.
IWS MODE	Toggle switch	Selects size of instruction word stack (IWS). Up position selects 8 words. Middle position selects 12 words. Down position selects 2 words.
8 WORD, 2 WORD	Indicators	Light to show respective IWS words selected. Neither indicator lighted denotes a 12-word IWS selection.
MEMORY MODE	Toggle switch	Selects a parity or single-error correction double-error detection (SECDED) mode. Changing the position of this switch requires CM to be rewritten. Up position selects parity mode. Down position selects SECDED mode.
PARITY	Indicator	Lights to show parity mode selection.

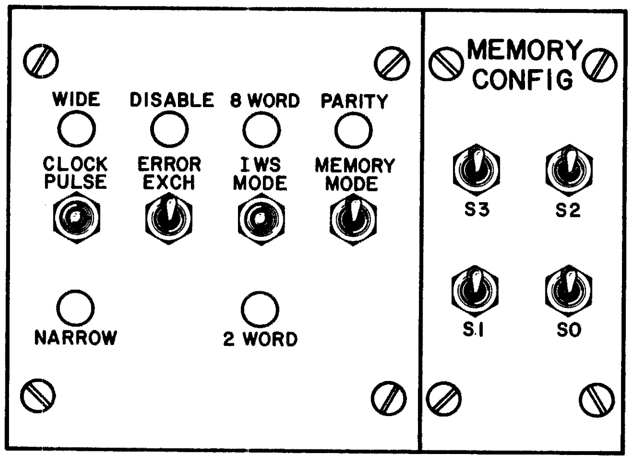


Figure 3-2. Module at CP Locations A1 through A3 (Model 175)

TABLE 3-3. MODEL 175 MEMORY SELECTION SCHEME

Central Memory Size	Range of Address	Normal Operation Switch Positions †				Reconfigured Operation Switch Positions †				
		Memory Config. Switches S1	S2	S3	S4	Bad Quad	Memory Config. Switches S1	S2	S3	S4
65K	0-177777	1	0	0	0	0	No Reconfiguration			
98K	0-277777	1	1	0	0	0	No Reconfiguration			
						1	1	0	0	0
131K	0-377777	1	1	0	0	0	0	1	0	0
						1	1	0	0	0
196K	0-577777	1	1	1	0	0	0	1	1	0
						1	1	0	1	0
						2	1	1	0	0
262K	0-777777	1	1	1	1	0	0	1	1	1
						1	1	0	1	1
						2	1	1	0	1
						3	1	1	1	0

†Switches generate a 1 when in the up position and a 0 when in the down position.

CENTRAL MEMORY CONTROL CHASSIS MODELS 172 THROUGH 174

The CMC chassis contains modules with control switches at locations 4N24, 4N25, 4B34, 4R39, and 4R40.

Module Locations 4N24 and 4N25

These modules (Figure 3-3) contain switches for the selection of normal and reconfigured CM and SECDED or parity mode operation. Tables 3-4 and 3-5 list the switches and their functions. Table 3-6 lists the switch settings for normal and reconfigured CM operation.

All CM quadrants, for a particular CM size, are available for use by model 172 through 174 systems when the address range control and bad quadrant code switches are set for normal operation as shown in Table 3-6.

If one of the CM quadrants of CSU-0 or CSU-1 becomes defective, the CM may be reconfigured to operate without the quadrant. The reconfiguration is performed by determining the defective CSU quadrant and then setting the address range control and bad quadrant code switches for that quadrant to the reconfigured operation switch positions shown in Table 3-6. Any one CSU-0 or CSU-1 quadrant may be reconfigured at one time, except quadrant 0 of the 32K CM. The switches accomplish the logical reconfiguration by manipulating the CM upper address bits.

The switches are always active.

Module Location 4B34

This module (Figure 3-3) is the same as the module at 4C35 of model 175. The module provides the selection of data parity from the ECS controller, if the controller has parity enhancement. Only the top switch of the module is used. The switch selects parity enhancement in the down position. The switch must be in the up position when an ECS controller without parity enhancement is used. The up position may also be used to disable parity enhancement, if it is available but not used.

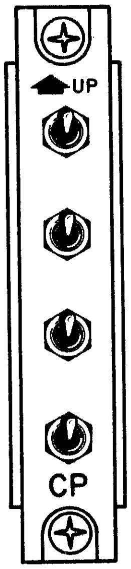


Figure 3-3. Module at CMC Locations 4N24, 4N25, and 4B34 (Models 172 through 174) and 4C35 (Model 175)

Module Locations 4R39 and 4R40

These modules are the same as the modules at 4B18 and 4B19 of model 175. The modules (Figure 3-4) contain clock switches and indicators. Table 3-7 lists the switches and indicators and their functions.

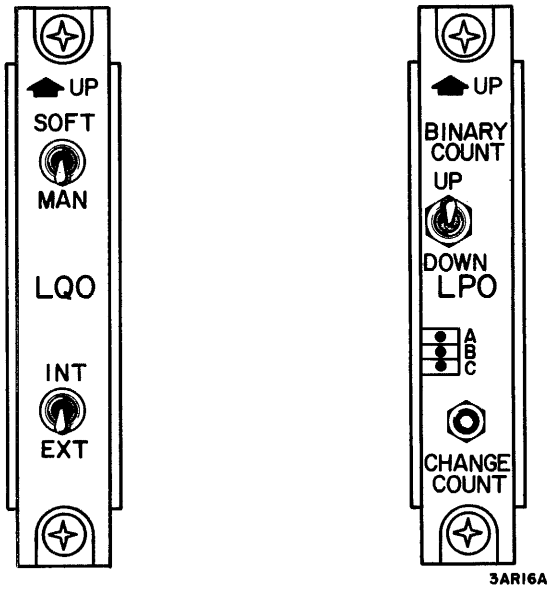


Figure 3-4. Module at CMC Locations 4R39 and 4R40 (Models 172 through 174) and 4B18 and 4B19 (Model 175)

TABLE 3-4. MODELS 172 THROUGH 174 CMC SWITCHES AT MODULE LOCATION 4N24

Panel Location	Description	Function
Top	Toggle switch	Address range control switch 1. Up position selects 0. Down position selects 1.
Next-to-top	Toggle switch	Address range control switch 2. Up position selects 0. Down position selects 1.
Next-to-bottom	Toggle switch	Address range control switch 3. Up position selects 0. Down position selects 1.
Bottom	Toggle switch	Address range control switch 4. Up position selects 0. Down position selects 1.

TABLE 3-5. MODELS 172 THROUGH 174 CMC SWITCHES AT MODULE LOCATION 4N25

Panel Location	Description	Function
Top	Toggle switch	Bad quadrant code switch 5. Up position selects 0. Down position selects 1.
Next-to-top	Toggle switch	Bad quadrant code switch 6. Up position selects 0. Down position selects 1.
Next-to-bottom	Toggle switch	Bad quadrant code switch 7. Up position selects 0. Down position selects 1.
Bottom	Toggle switch	Selects memory mode. Up position selects parity mode. Down position selects SECDED mode.

TABLE 3-6. MODELS 172 THROUGH 174 MEMORY SELECTION SCHEME

Central Memory Size	Range of Address	Normal Operation Switch Positions †							Reconfigured Operation Switch Positions †								
		Adrs Range Control Switch				Bad Quad Code Switch			Reconfigured Selection		Adrs Range Control Switch				Code Switch		
		1	2	3	4	5	6	7	CSU	Bad Quad	1	2	3	4	5	6	7
32K	0-077777	0	0	0	1	1	1	1	0	0	No Reconfiguration						
49K	0-137777	0	0	1	0	1	1	1	0	0	0	0	0	1	0	0	0
									0	1	0	0	0	1	0	0	1
65K	0-177777	0	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0
										1	0	0	0	1	0	0	1
98K	0-277777	0	1	1	0	1	1	1	0	0	0	0	1	1	0	0	0
										1	0	0	1	1	0	0	1
										2	0	0	1	1	0	1	0
131K	0-377777	0	1	1	1	1	1	1	0	0	0	1	1	0	0	0	0
										1	0	1	1	0	0	0	1
										2	0	1	1	0	0	1	0
										3	0	1	1	0	0	1	1
196K	0-577777	1	1	1	0	1	1	1	0	0	0	1	1	1	0	0	0
										1	0	1	1	1	0	0	1
										2	0	1	1	1	0	1	0
										3	0	1	1	1	0	1	1
									1	0	0	1	1	1	1	0	0
										1	0	1	1	1	1	0	1
262K	0-777777	1	1	1	1	1	1	1	0	0	1	1	1	0	0	0	0
										1	1	1	1	0	0	0	1
										2	1	1	1	0	0	1	0
										3	1	1	1	0	0	1	1
									1	0	1	1	1	0	1	0	0
										1	1	1	1	0	1	0	1
										2	1	1	1	0	1	1	0
										3	1	1	1	0	1	1	1

†Switches generate a 0 when in the up position and a 1 when in the down position.

TABLE 3-7. CMC CONTROLS AND INDICATORS AT MODULE LOCATIONS 4R39 and 4R40 (MODELS 172 THROUGH 174) AND 4B18 AND 4B19 (MODEL 175)

Panel Nomenclature	Description	Function
INT EXT	Toggle switch	Selects internal or external clock source. Internal source is required for clock-margin checks. External source is required for use with ECS.
SOFT MAN	Toggle switch	Selects software or manual control of the clock frequency margins. Software control is described by the status and control register bits 141 through 143 in section 2. Manual control is described by the following switches.
CHANGE COUNT	Push-button switch	Permits the clock to be manually incremented to a fast, slow, or normal operating frequency, when the SOFT/MAN switch is set to MAN. Normal operation requires the clock to be set at the normal operating frequency as indicated by the light-emitting diodes A, B, and C.
A B C	Indicators	Indicate a binary count. C is bit 0, B is bit 1, and A is bit 2. The count for normal clock operating frequency is 3, where A is not lighted and B and C are lighted. Further use of these indicators is described in the hardware maintenance manuals listed in the system publication index.
BINARY COUNT UP/DOWN	Toggle switch	Selects an increment or decrement of the binary count, changed with the CHANGE COUNT switch.

CENTRAL MEMORY CONTROL CHASSIS MODEL 175

The CMC chassis contains modules with controls and indicators at module locations 4C35, 4B18, and 4B19.

Module Location 4C35

This module (Figure 3-3) is the same as the module at 4B34 of models 172 through 174. The module provides the selection of data parity from the ECS controller, if the controller has parity enhancement. Only the top switch of the module is used. The switch selects parity enhancement in the down position. The switch must be in the up position when an ECS controller without parity enhancement is used. The up position may also be used to disable parity enhancement, if parity enhancement is available but not used.

Module Locations 4B18 and 4B19

These modules are the same as the modules at 4R39 and 4R40 of models 172 through 174. The modules (Figure 3-4) contain clock switches and indicators. Table 3-7 lists switches, indicators, and their functions.

CENTRAL STORAGE UNIT CHASSIS

Each CSU contains one module (Figure 3-5) at location Q30. The module has three two-position toggle switches for maintenance purposes. Table 3-8 lists the switches and their functions.

The switches are always active.

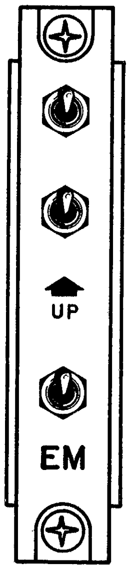


Figure 3-5. Module at CSU Location Q30

TABLE 3-8. CSU CHASSIS SWITCHES AT MODULE LOCATION Q30

Panel Location	Description	Function
Top	Toggle switch	Up position disables the eight CSU memory banks. Down position enables the banks.
Middle	Toggle switch	Up position disables address parity detection in the eight CSU memory banks. Down position enables the address parity detection.
Bottom	Toggle switch	Up position provides constant clear in the eight CSU memory banks. Middle position disables the internal master clear. Down position forces internal master clear.

INPUT/OUTPUT CONNECTOR PANEL

The I/O connector panel (Figure 3-6) is located on the end of bay 1, next to the hinged side of the chassis. The panel contains 12 PARITY switches that enable (ON position) or disable (OFF position) input parity checking for their respective channels. If any of the PARITY switches are set to the OFF position, the parity switches for the peripherals of the corresponding channels must also be set to the OFF position.

Parity for optional channels is controlled by additional PARITY switches located on an optional I/O connector panel, opposite the basic system panel.

The PARITY switches for the basic and optional I/O channels are always active.

PERIPHERAL PROCESSOR SUBSYSTEM CHASSIS

The PPS contains controls/indicators at 10 module locations. The modules and locations are the same in PPS-0 and PPS-1.

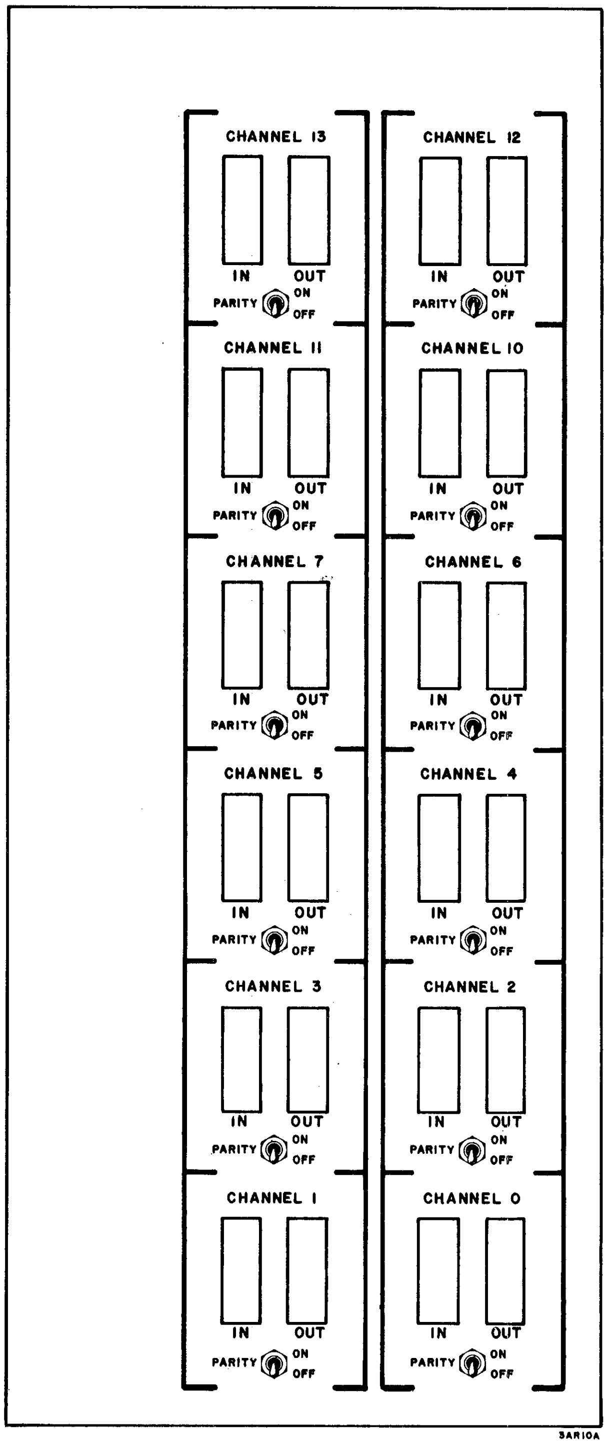


Figure 3-6. I/O Connector Panel Controls

PPS Module at Location J40

The PPS module at location J40 (Figure 3-7) is a switch pack that has four two-position toggle switches. The switches select the contents of any of the PPS P registers for display on the light module in PPS location H33.

The switches also define which PP is enabled for status bits 125 and 126 when status and control register status bit 124 is not set.

The switches form a binary number with the bottom switch being bit 0 and the top switch bit 3. Table 3-9 lists the switches and their functions.

The switches are always active.

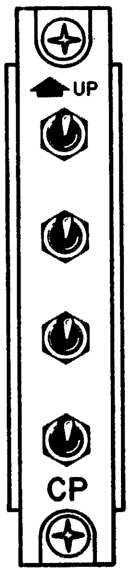


Figure 3-7. Module at PPS Location J40

TABLE 3-9. PPS CHASSIS MODULE SWITCHES AT MODULE LOCATION J40

Panel Location	Description	Function
Top	Toggle switch	Up position enables bit 3. Down position disables bit.
Next-to-top	Toggle switch	Up position enables bit 2. Down position disables bit.
Next-to-bottom	Toggle switch	Up position enables bit 1. Down position disables bit.
Bottom	Toggle switch	Up position enables bit 0. Down position disables bit.

PPS Modules at Locations R36 and P38

The PPS module shown in Figure 3-8 is a keyboard input receiver pack that has one two-position toggle switch. The module is used in PPS locations R36 and P38. The module at R36 enables (switch up) or disables (switch down) the keyboard of display station 0. The module at P38 enables (switch up) or disables (switch down) the keyboard of display station 1, if station 1 is installed. The module switches may be set to enable either or both keyboards at the same time. Normal operation is to enable only one keyboard at a time to prevent the possibility of making entries from both keyboards at the same time.

The switch on each module is always active.

PPS with Light Modules

The light modules (Figures 3-9 through 3-15) have two columns of nine, red, light-emitting diodes. The diodes indicate the condition of certain status and control bits of the status and control register. Each diode represents one bit in the register; it lights when the corresponding register bit is set. A push-button switch below the diodes permits testing of the diodes.

Figures 3-9 through 3-15 show the light modules at locations F19, F22, F25, F28, H33, F39, and F42 and give the bit numbers, status or control use, and bit descriptions of the light-emitting diodes.

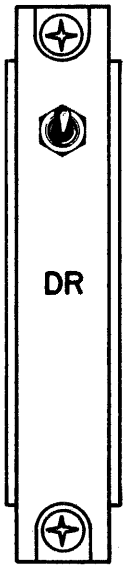
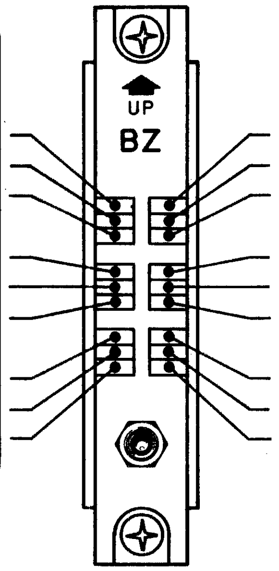


Figure 3-8. Module at PPS Locations R36 and P38

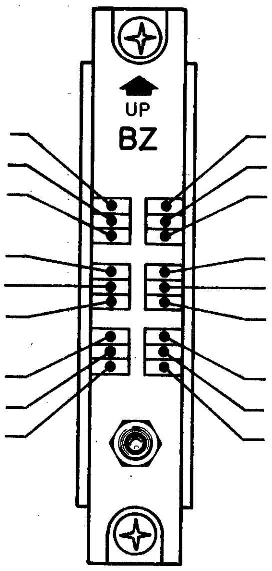
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
8	10	S	CSU-0 FAULT
9	11	S	CSU-1 FAULT
10	12	S	ERROR IN SECOND PPS
			NOT USED
12	14	S	CP-0 P RGTR PE
13	15	S	CP-1 P RGTR PE
14	16	S	PPO MEMORY PE
15	17	S	PP1 MEMORY PE
			NOT USED



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
0	0	S	READ PYRAMID PE
1	1	S	CSU-0 ADRS PE
2	2	S	CSU-1 ADRS PE
3	3	S	SECDED ERROR
183	267	S	DOUBLE ERROR
5	5	S	CMC PE
6	6	S	PE FROM EXT CHAN
7	7	S	PE FROM EXT PP
			NOT USED

Figure 3-9. Module at PPS Location F19

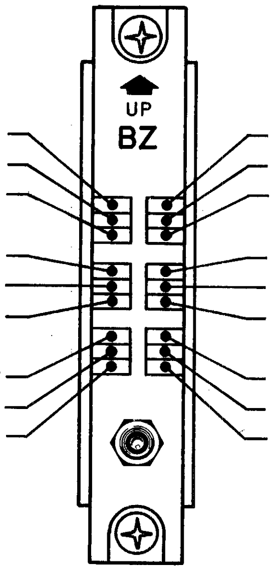
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
24	30	S	CHANNEL 0 PE
25	31	S	CHANNEL 1 PE
26	32	S	CHANNEL 2 PE
27	33	S	CHANNEL 3 PE
28	34	S	CHANNEL 4 PE
29	35	S	CHANNEL 5 PE
30	36	S	CHANNEL 6 PE
31	37	S	CHANNEL 7 PE
			NOT USED



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
16	20	S	PP2 MEMORY PE
17	21	S	PP3 MEMORY PE
18	22	S	PP4 MEMORY PE
19	23	S	PP5 MEMORY PE
20	24	S	PP6 MEMORY PE
21	25	S	PP7 MEMORY PE
22	26	S	PP8 MEMORY PE
23	27	S	PP9 MEMORY PE
			NOT USED

Figure 3-10. Module at PPS Location F22

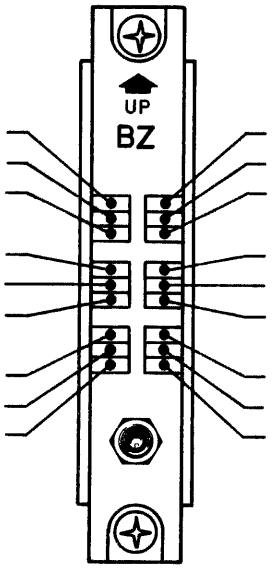
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
40	50	S	SYNDROME BIT 0
41	51	S	SYNDROME BIT 1
42	52	S	SYNDROME BIT 2
43	53	S	SYNDROME BIT 3
44	54	S	SYNDROME BIT 4
45	55	S	SYNDROME BIT 5
46	56	S	SYNDROME BIT 6
47	57	S	SYNDROME BIT 7
118	166	C	INHIBIT SE REPORT



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
32	40	S	CHANNEL 10 PE
33	41	S	CHANNEL 11 PE
34	42	S	CHANNEL 12 PE
35	43	S	CHANNEL 13 PE
36	44	S	MAINS POWER FAILURE
37	45	S	SHUTDOWN IMMINENT
38	46		NOT USED
39	47		NOT USED
			NOT USED

Figure 3-11. Module at PPS Location F25

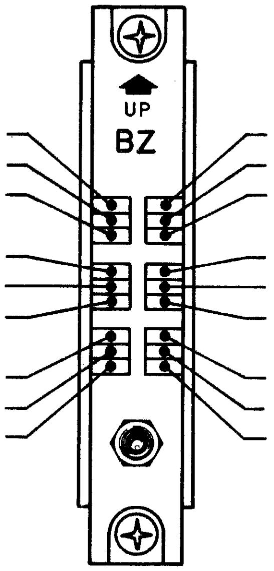
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
56	70	S	BRKPT PORT CODE BIT 0
57	71	S	BRKPT PORT CODE BIT 1
58	72	S	BRKPT FCTN CODE BIT 0
59	73	S	BRKPT FCTN CODE BIT 1
			NOT USED
			NOT USED
			NOT USED
			NOT USED
140	214	S	NOT USED



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
48	60	S	SYNDROME ADRS BIT 0
49	61	S	SYNDROME ADRS BIT 1
50	62	S	SYNDROME ADRS BIT 2
51	63	S	SYNDROME ADRS BIT 16
52	64	S	SYNDROME ADRS BIT 17
53	65	S	SYNDROME ADRS BIT 3
54	66	S	PE PORT CODE BIT 0
55	67	S	PE PORT CODE BIT 1
139	213	S	CMC ADRS/DATA PE

Figure 3-12. Module at PPS Location F28

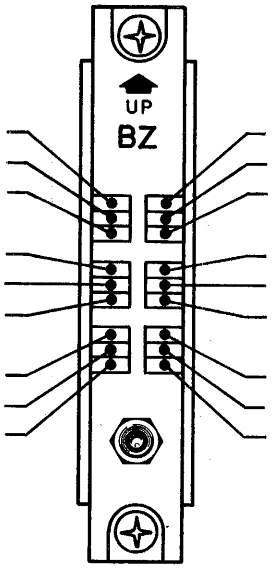
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
69	105	S	PPS P RGTR BIT 9
70	106	S	PPS P RGTR BIT 10
71	107	S	PPS P RGTR BIT 11
72	110	S	PP CODE BIT 0
73	117	S	PP CODE BIT 1
74	112	S	PP CODE BIT 2
75	113	S	PP CODE BIT 3
76	114	S	PPS BRKPT BIT
77	115	S	CMC BRKPT MATCH



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
60	74	S	PPS P RGTR BIT 0
61	75	S	PPS P RGTR BIT 1
62	76	S	PPS P RGTR BIT 2
63	77	S	PPS P RGTR BIT 3
64	100	S	PPS P RGTR BIT 4
65	101	S	PPS P RGTR BIT 5
66	102	S	PPS P RGTR BIT 6
67	103	S	PPS P RGTR BIT 7
68	104	S	PPS P RGTR BIT 8

Figure 3-13. Module at PPS Location H33

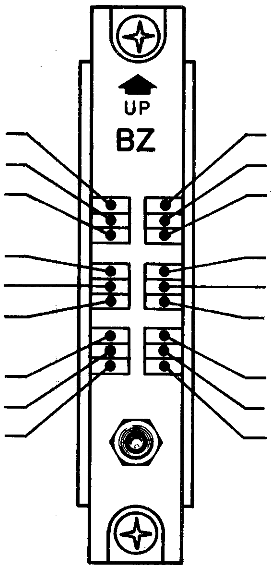
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
93	135		NOT USED
94	136		NOT USED
95	137	C	STOP ON PP MEMORY PE
192	300	S	CP-0 STOPPED
193	301	S	CP-1 STOPPED
194	302	S	ECS IN PROGRESS FLAG
195	303	S	MONITOR FLAG CP-0
196	304	S	MONITOR FLAG CP-1
			NOT USED



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
84	124	C	ALL PPs 500 NSEC MAJ CYCLE
85	125	C	INHIBIT CMC REQUEST
86	126		NOT USED
			NOT USED
			NOT USED
			NOT USED
			NOT USED
			NOT USED
			NOT USED

Figure 3-14. Module at PPS Location F39

BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
144	220	S	RVM ADRS BIT 0 STATUS
145	221	S	RVM ADRS BIT 1 STATUS
146	222	S	RVM ADRS BIT 2 STATUS
147	223	S	RVM ADRS BIT 3 STATUS
148	224	S	RVM ADRS BIT 4 STATUS
149	225	S	RVM ADRS BIT 5 STATUS
150	226	S	RVM ADRS HI/LO
151	227	S	RVM ALL/ONE
			NOT USED



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
120	170	C	PP SELECT CODE BIT 0
121	171	C	PP SELECT CODE BIT 1
122	172	C	PP SELECT CODE BIT 2
123	173	C	PP SELECT CODE BIT 3
124	174	C	PP SELECT AUTO/MNL MODE
			NOT USED
			NOT USED
152	230	C	CLK PULSE NARROW
153	231	C	CLK PULSE WIDE

Figure 3-15. Module at PPS Location F42

OPERATING PROCEDURES

Prior to program operation and keyboard control, the system controls should be checked, a deadstart program selected, and the system deadstarted. After the system is put into operation, the display station keyboard provides manual control of the system and entry of data or instructions into the system under program control.

CONTROL CHECKS

The following list suggests normal operating positions of the system control switches. The switches should be checked and their intended use confirmed before deadstarting the system.

1. Deadstart panel

Switch	Position
2^0 through 2^{11} by 1 through 20 matrix	User's discretion, according to selected deadstart program
CEJ/MEJ	User's discretion
PPS-0/PPS-1	User's discretion
PPS SELECT	Down, all switches
SWEEP/LOAD/DUMP	LOAD
DEAD START	Down

2. Module locations 4A1 through 5A3 (model 175)

Switch	Position
S0 through S3	Refer to Table 3-1
CLOCK PULSE	Middle
ERROR EXCH	Down
IWS MODE	Middle
MEMORY MODE	Down

3. Module location 4N24 (models 172 through 174)

Switch	Position
Top	Refer to Table 3-2
Next-to-top	Refer to Table 3-2
Next-to-bottom	Refer to Table 3-2
Bottom	Refer to Table 3-2

4. Module location 4N25 (models 172 through 174)

Switch	Position
Top	Refer to Table 3-2
Next-to-top	Refer to Table 3-2
Next-to-bottom	Refer to Table 3-2
Bottom	Down

5. Module locations 4B34 (models 172 through 174) and 4C35 (model 175)

Switch	Position
TOP	Depends upon ECS parity requirements
All others	Unused

6. Module locations 4R39 (models 172 through 174) and 4B18 (model 175)

Indicator	Condition
A	Not lighted
B	Lighted
C	Lighted

7. Module locations 4R40 (models 172 through 174) and 4B19 (model 175)

Switch	Position
INT/EXT	EXT (if ECS is used)

8. CSU module location Q30, all switches down

9. PPS module location J40, all switches down

10. PPS module locations R36 and P38, switch up on one or both modules at user's discretion

11. I/O connector panel, PARITY switches in ON position for channels having equipment attached which provide parity

DEADSTART PROGRAM SELECTION

Refer to the system operator's guide or installation handbook to select a deadstart program of 16 words or less for the deadstart panel. The deadstart program is normally a load routine which loads a larger program from input equipment.

DEADSTART

The system deadstart panel provides a load, sweep, or dump mode of operation. These modes are initiated from the deadstart panel.

Load Mode

Load programs and data into the computer system as follows:

1. Set SWEEP/LOAD/DUMP switch to LOAD.
2. Set 2^0 through 2^{11} by 1 through 20 (octal) toggle switch matrix according to selected deadstart program. (Bits are set when switches are in the up position.)
3. Check that PPS-1/PPS-0, and PPS SELECT switches select the desired PP to be PP-0.
4. Set DEAD START switch to ON position, momentarily.

Results of steps 1 through 4:

1. Assign data channels 0 through 11_8 to corresponding PPs in each PPS.
2. Send a master clear to all I/O channels. With the exception of the deadstart panel, a master clear selects the first data channel converter on each channel and resets bits 80 through 95 and 120 through 127 of the status and control register. Master clear sets all channels to the active and empty condition, ready for inputs.
3. Set all PPs to the input (712) instruction.
4. Clear the P registers and set the A registers to 10000 in all PPs.
5. Transmit a zero word that is followed by the 16 words from the toggle switches into PPM locations 0_8 through 20_8 of PP-0. Channel 0 is then disconnected, clearing word 21_8 of PP0 and causing PP0 to start execution with the instruction at location 0001.

Sweep Mode

Sweep mode is a maintenance function useful in checking PPM operation. Initiate this mode as follows:

1. Set SWEEP/LOAD/DUMP switch to SWEEP.
2. Set DEAD START switch to ON position, momentarily. (DEAD START switch may be left on for syncing purposes.)

Results of steps 1 and 2:

1. Set all PPs to load (505) instruction.
2. Clear all PP P registers and start the P registers counting.
3. Cause each PP to sweep through its PPM, reading the content of each location, without executing instructions.

Dump Mode

Dump mode provides copying the content of CM to an external storage device in case of a computer malfunction. Initiate dump program in PP-0 as follows:

1. Set SWEEP/LOAD/DUMP switch to DUMP.
2. Set DEAD START switch to ON position, momentarily.

Results of steps 1 and 2:

1. Set all PPs to output (732) instruction.
2. Send a master clear on all channels, except channel 0.
3. Hold channel 0 active and empty.
4. Assign each PP to its corresponding I/O channel.
5. Clear all A and P registers.
6. Cause all PPs to sense the empty and active conditions of their assigned channels, output the content of their address 0000, set their channels to full, and wait for an empty condition.

INSTRUCTION DESCRIPTIONS 4

This section describes the central processor (CP) and peripheral processor (PP) instructions. Some differences exist in the CP instructions because of model differences. The instruction differences are identified with the applicable model numbers. PP instructions are identical for all models. (Other programming and system error response information is in section 5.)

CP and PP instruction codes and page numbers are listed in indexes on the inside front cover for quick reference.

CENTRAL PROCESSOR INSTRUCTIONS

The CP instructions are in two categories, those causing computation and those causing storage references or program branching. The instructions causing computation are generally executed in a fixed amount of time after they have issued. Instructions involving storage references for operands or program branching cannot be precisely timed. Careful coding of critical program loops can produce substantial improvements in execution time. Detailed timing information is in appendix A to allow a complete analysis of the situations warranting the programming effort.

INSTRUCTION FORMATS

Program instruction words are divided into 15-bit fields called parcels. The first parcel (parcel 0) is the highest-order 15 bits of the 60-bit word. The second, third, and fourth parcels (parcels 1, 2, and 3) follow in order. An instruction may occupy one, two, or four parcels, depending upon the type of instruction. When an instruction occupies two parcels, it must occupy two parcels within the same program word. Possible parcel arrangements are illustrated in Figure 4-1. Instruction designators are listed and defined in Table 4-1.

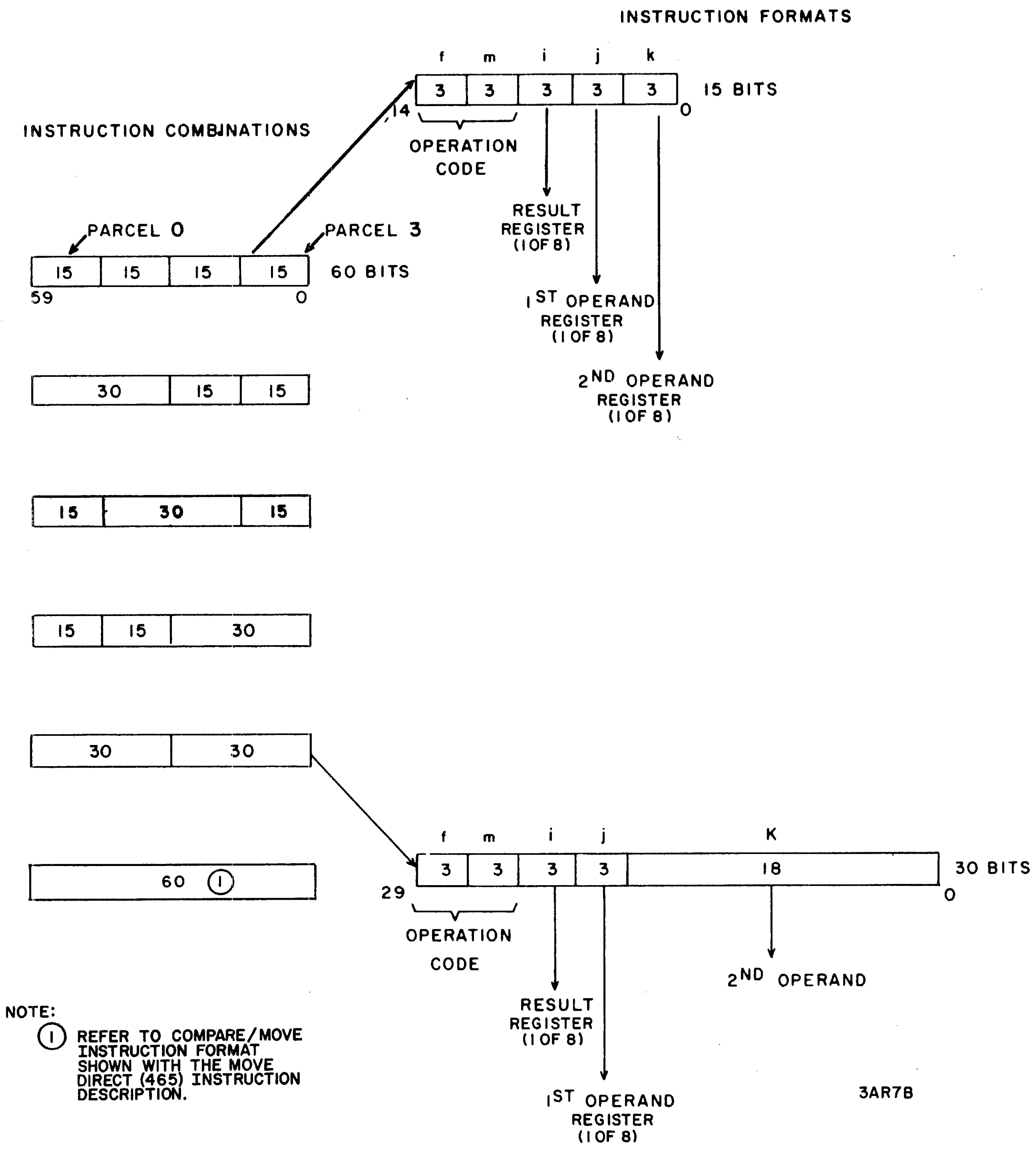


Figure 4-1. Parcel Instruction Arrangements

A program word may be filled with a one-parcel pass instruction or an instruction acting as a two-parcel pass instruction. These instructions are used to fill a program word when necessary to place a particular instruction in the first parcel of a program word or to avoid starting a two-parcel instruction in the fourth parcel of a program word. Pass instructions may also be used for branch entry points because a branch instruction destination address must begin with a new word. One-parcel pass instructions are 4600XX through 463XX. Instructions 60XXX through 62XXX may be used as two-parcel pass instructions by setting the i instruction designator to zero.

TABLE 4-1. CENTRAL PROCESSOR INSTRUCTION DESIGNATORS

Designator	Use
fm	6-bit instruction code
fmi	9-bit instruction code
i	3-bit code specifying one of eight registers
j	3-bit code specifying one of eight registers
jk	6-bit code specifying amount of shift or mask
k	3-bit code specifying one of eight registers
K	18-bit operand or address
x	Unused designator
A	One of eight 18-bit address registers
B	One of eight 18-bit index registers; B0 is fixed and equal to zero
X	One of eight 60-bit operand registers
( )	Content of a register or location
	<u>Compare/Move (Models 172, 173, and 174)</u>
C1	Offset (character address) of the first character in the first word of the source field
C2	Character address of the first character in the first word of the result field

TABLE 4-1. CENTRAL PROCESSOR INSTRUCTION DESIGNATORS (Cont'd)

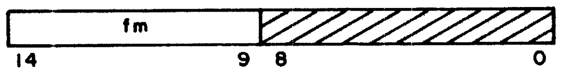
Designator	Use
K1	18-bit address indicating the central memory location of the first (leftmost) character of the source field
K2	18-bit address indicating the central memory location of the first (leftmost) character of the result field
LL	Lower 4 bits of the field length (character count) for a move or compare instruction; used with LU to specify field length
LU	Upper 9 bits of the field length (character count) for indirect move instruction or the upper 3 bits for direct instructions; used with LL to specify field length

INSTRUCTION DESCRIPTIONS

This part of the manual describes the CP instructions. The instructions are described separately and in numerical order.

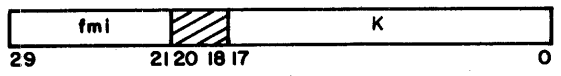
Shaded areas, like those in the following 00xxx and 010xK instruction formats, indicate unused bits. The unused bits are ignored by the CP.

00xxx Error Exit to MA or Program Stop



The CEJ/MEJ switch determines which functions this instruction can perform. When the switch is in the DISABLE position, the system has no central exchange or monitor exchange jump capability so this instruction stops the CP. When the switch is in the ENABLE position, this instruction causes an error exit response that is the same as an illegal instruction. (Refer to Error Response in section 5.)

010xK Return Jump to K



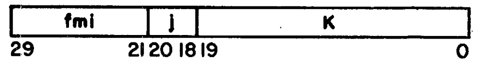
This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction writes a special word into central memory (CM) at relative address K. The current program sequence is then terminated by a jump to address K plus 1. The word stored in memory contains a jump instruction which causes an unconditional jump to the address of this return jump instruction plus 1. In model 175, a return jump out of the instruction word stack (IWS) voids the stack.

This instruction calls a subroutine and inserts execution of the subroutine between execution of this instruction word and the following instruction word. Instructions appearing after the return jump instruction in the instruction word are not executed. The called subroutine exit must be at address K. The called subroutine entrance address must be K plus 1.

This instruction stores a 60-bit word at address K in memory. The upper half of this word contains an unconditional jump instruction (0400) with an address which is equal to the current program address plus 1. The lower half of the stored word is all zeros. The octal digits in the stored word then appear as illustrated with the x field indicating the location of the current program address plus 1.

K	0400x xxxxx 00000 00000	Subroutine exit
K + 1	yyyyy yyyyy yyyyy yyyyy	Subroutine entrance

011jK Block Copy (Bj) + K Words from ECS to CM



This is a 30-bit instruction that uses bits 0 through 17 as operand K. The instruction reads a block of 60-bit words from consecutive addresses in extended core storage (ECS) to consecutive addresses in CM. The consecutive addresses are relative addresses that begin at X0 in ECS and A0 in CM. The length of the block of words read is the sum of the content of Bj plus K.

Three of the parameters for this instruction reside in operating registers (A0, X0, and Bj). The contents of these registers are not altered by the execution of this instruction.

When bit 23 of X0 and bit 23 of the FLE register in the CP are set, a flag register operation is performed in the ECS controller in place of a block copy. The flag register operation provides information about current or previous programs by performing a ready/set, selective set, status, or selective clear function. For additional flag register information, refer to the ECS Hardware Reference Manual listed in the system publication index.

This instruction moves a quantity of data from ECS into CM as quickly as possible. All other activity, except peripheral processor subsystem (PPS) word requests, is stopped during the block transfer of data. In model 174, activity in the second CP continues, except for exchange jumps. In simultaneous ECS requests, CP-0 has priority over CP-1.

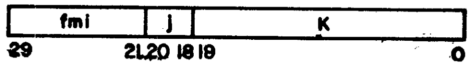
All instructions issued prior to this instruction are executed to completion prior to the beginning of data transfer. No further instructions are issued until this block transfer is completed. As a result, the data flow from ECS to CM proceeds at a rate up to one 60-bit word each 100 nanoseconds, once the transfer of data is started.

The maximum length of a block transfer is 131K 60-bit words and is determined by the addition of the signed integers in Bj and K. Both the CP and the ECS check the result of the addition, performed in an 18-bit ones-complement mode. The result is treated as an 18-bit integer. A zero or negative result executes this instruction as a pass instruction.

The instruction must be located in parcel 0 of the instruction word. The instruction is illegal if ECS is not present or if the instruction does not reside in parcel 0 of the instruction word. (Refer to Illegal Instructions in section 5.)

The normal exit for this instruction is to the content of the P register plus 1 (P plus 1). An exit to the lower 30 bits of the instruction word occurs on an error condition. The error condition can be a central memory control (CMC) double error or any of the following parity errors: CP to ECS coupler, CP to CMC address, CMC data, ECS bank, ECS controller data, or ECS controller address.

012jK Block Copy (Bj) + K Words from CM to ECS



This is a 30-bit instruction that uses bits 0 through 17 as operand K. The instruction reads a block of 60-bit words from consecutive addresses in CM to consecutive addresses in ECS. The consecutive addresses are relative addresses that begin at A0 in CM and X0 in ECS. The length of the block of words read is the sum of the content of Bj plus K.

Three of the parameters for this instruction reside in operating registers (A0, X0, and Bj). The contents of these registers are not altered by the execution of this instruction.

When bit 23 of X0 and bit 23 of the FLE register in the CPU are set, a flag register operation is performed in the ECS controller in place of a block copy. The flag register operation provides information about current or previous programs by performing a ready/set, selective set, status, or selective clear function. For additional flag register information, refer to the ECS Hardware Reference Manual listed in the system publication index.

This instruction moves a quantity of data from CM into ECS as quickly as possible. All other activity, except PPS word requests, is stopped during the block transfer of data. In model 174, activity in the second CP continues, except for exchange jumps. In simultaneous ECS requests, CP-0 has priority over CP-1.

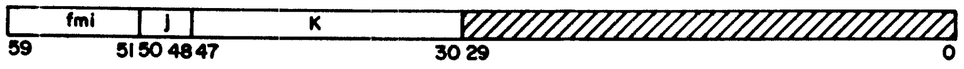
All instructions issued prior to this instruction are executed to completion prior to the beginning of data transfer. No further instructions are issued until this block transfer is completed. As a result, the data flow from CM to ECS proceeds at a rate up to one 60-bit word each 100 nanoseconds, once the actual transfer of data is started.

The maximum length of a block transfer is 131K 60-bit words and is determined by the addition of the signed integers in Bj and K. Both the CP and the ECS check the result of the addition, performed in an 18-bit ones-complement mode. The result is treated as an 18-bit integer. A zero or negative result executes the instruction as a pass instruction.

The instruction must be located in parcel 0 of the instruction word. The instruction is illegal if ECS is not present or if the instruction does not reside in parcel 0 of the instruction word. (Refer to Illegal Instructions in section 5.)

The normal exit for this instruction is P plus 1. An exit to the lower 30 bits of the instruction occurs on an error condition. The error condition can be a CMC double error or any of the following parity errors: CP to ECS coupler, CP to CMC address, CMC data, ECS bank, ECS controller data, or ECS controller address.

013jK Central Exchange Jump to (Bj) + K (Monitor Flag Set)



This is a 60-bit instruction in which bits 30 to 47 are used as operand K. The starting address for the exchange jump is the 18-bit result formed by adding K to the content of the Bj register. This starting address is an absolute address. At the end of the exchange jump, the monitor flag clears.

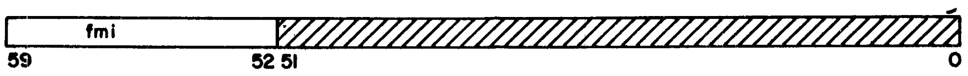
This form of the 013 instruction is used by the monitor program only. The monitor program uses this instruction to exchange jump to one of a number of possible object program exchange packages. A selected object program exchange package returns to this same area of CM and resumes the monitor program when its execution interval has been completed (refer to the following alternate form of 013 instruction).

This instruction has priority over PPS exchange jump requests. If a PPS exchange request occurs simultaneously with the execution of this instruction, the request waits until the central exchange has completed. Error exit exchange requests, if they occur, are processed before the central exchange jump executes.

The program address stored in the exchange package is advanced one count from the address of the instruction word. Therefore, the program continues at parcel 0 of the following instruction word during the next execution interval for this exchange package.

This instruction is illegal if the CEJ/MEJ switch is in the DISABLE position or if the instruction does not reside in parcel 0 of the instruction word. (Refer to Illegal Instructions in section 5.)

013xx Central Exchange Jump to MA (Monitor Flag Not Set)



A central exchange jump instruction executed in this mode causes the current program sequence to terminate with an exchange jump to the monitor address (MA). This is an absolute address in CM and is generally not in the CM field for the current program. This mode makes no use of the j or k designators in the instruction. At the end of the exchange jump, the monitor flag sets.

This instruction allows switching from an object program to a monitor program. All operating register values, program addresses, and mode selections are preserved in this process so the object program may be continued at a later time. The program address in the object program exchange package is advanced one count from the address of the instruction word containing the exchange exit instruction. The monitor program normally resumes the object program at this address.

This instruction calls the system monitor program for PPS requests, library calls, storage assignments, and so on. The operating register values at the time of execution of this instruction allow parameter interchange between the object program and the monitor program.

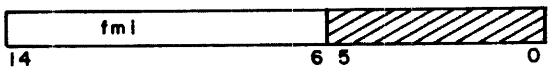
This instruction has priority over PPS exchange jump requests. If a PPS exchange request occurs simultaneously with the execution of this instruction, the request waits until the central exchange has completed. Error exchange requests, if they occur, are processed before the central exchange jump executes.

The program address stored in the exchange package is advanced one count from the address of the instruction word. Therefore, the program continues at parcel 0 of the following instruction word during the next execution interval for this exchange package unless the monitor program alters the exchange package.

This instruction is illegal if the CEJ/MEJ switch is in the DISABLE position or if the instruction does not reside in parcel 0 of the instruction word. (Refer to Illegal Instructions in section 5.)

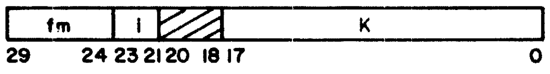
In model 174 with one CP in the monitor mode, the second CP cannot jump and waits until the monitor flag of the first CP clears.

014xx through 017xx Instructions



These instructions are defined as illegal. (Refer to Illegal Instructions in section 5.)

02ixK Jump to (Bi) + K

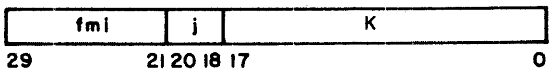


This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. In model 175, this instruction unconditionally voids the IWS. The instruction causes the current program sequence to terminate with a jump to address Bi plus K in CM.

This instruction allows computed branch point destinations. This is the only instruction in which a computed parameter can specify a program branch destination address. All other jump instructions have preassigned destination addresses.

The quantities in Bi and operand K are added in an 18-bit ones-complement mode. The result is treated as an 18-bit positive integer. This sum specifies the beginning address in CM for the new program sequence. The remaining instructions, if any, in the instruction word are not executed.

030jK Branch to K if (Xj) = 0



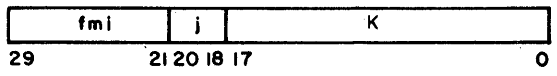
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The branch to address K occurs only on the following conditions. The current program sequence is continued for all other cases.

Jump to K if: (Xj) = 0000 0000 0000 0000 0000 (positive zero)
(Xj) = 7777 7777 7777 7777 7777 (negative zero)

This instruction is for branching on a zero result from either a fixed-point or a floating-point operation.

In model 175, a jump out of the IWS voids the stack.

031jK Branch to K if (Xj) ≠ 0



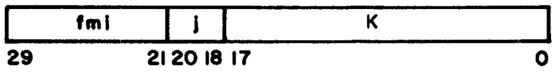
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The program sequence is continued only on the following conditions. The branch to address K occurs for all other cases.

Continue if: (Xj) = 0000 0000 0000 0000 0000 (positive zero)
(Xj) = 7777 7777 7777 7777 7777 (negative zero)

This instruction is for branching on a nonzero result from either a fixed-point or a floating-point operation.

In model 175, a jump out of the IWS voids the stack.

032jK Branch to K if (Xj) Positive



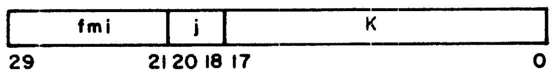
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The branch decision for this instruction is based on the value of the sign bit in Xj.

Jump to K if:	Bit 59 of Xj = 0	(positive)
Continue if:	Bit 59 of Xj = 1	(negative)

This instruction is for branching on a positive result from either a fixed-point or a floating-point operation.

In model 175, a jump out of the IWS voids the stack.

033jK Branch to K if (Xj) Negative



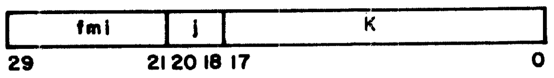
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The branch decision for this instruction is based on the value of the sign bit in Xj.

Jump to K if:	Bit 59 of Xj = 1	(negative)
Continue if:	Bit 59 of Xj = 0	(positive)

This instruction is for branching on a negative result from either a fixed-point or a floating-point operation.

In model 175, a jump out of the IWS voids the stack.

034jK Branch to K if (Xj) in Range



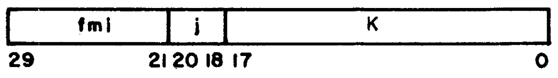
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The program sequence is continued only on the following conditions. The branch to address K occurs for all other cases.

Continue if: (Xj) = 3777 xxxx xxxx xxxx xxxx (positive overflow)
(Xj) = 4000 xxxx xxxx xxxx xxxx (negative overflow)

This instruction is for branching on a floating-point quantity within the floating-point range. The value of the coefficient is ignored in making this branch test. An underflow quantity is considered in range for purposes of this branch test.

In model 175, a jump out of the IWS voids the stack.

035jK Branch to K if (Xj) Out of Range



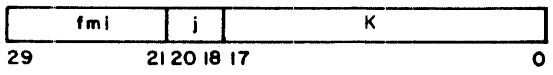
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The branch to address K occurs only on the following conditions. The current program sequence is continued for all other cases.

Jump to K if: (Xj) = 3777 xxxx xxxx xxxx xxxx (positive overflow)
(Xj) = 4000 xxxx xxxx xxxx xxxx (negative overflow)

This instruction is for branching on a floating-point quantity which is not in the floating-point range. The value of the coefficient is ignored in making this branch test. An underflow quantity is considered in range for purposes of this branch test.

In model 175, a jump out of the IWS voids the stack.

036jK Branch to K if (Xj) Definite



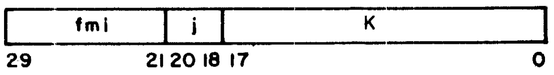
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The program sequence is continued only on the following conditions. The branch to address K occurs for all other cases.

Continue if:	(Xj) = 1777 xxxx xxxx xxxx xxxx (positive indefinite)
	(Xj) = 6000 xxxx xxxx xxxx xxxx (negative indefinite)

This instruction is for branching on a floating-point quantity which may be out of range but is still defined. The value of the coefficient is ignored in making this branch test. An overflow quantity or an underflow quantity is considered defined for purposes of this branch test.

In model 175, a jump out of the IWS voids the stack.

037jK Branch to K if (Xj) Indefinite



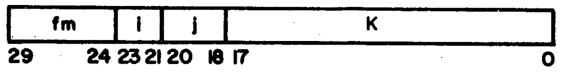
This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of the Xj register. The branch to address K occurs only on the following conditions. The current program sequence is continued for all other cases.

Jump to K if:	(Xj) = 1777 xxxx xxxx xxxx xxxx (positive indefinite)
	(Xj) = 6000 xxxx xxxx xxxx xxxx (negative indefinite)

This instruction is for branching on a floating-point quantity which is not defined. The value of the coefficient is ignored in making this branch test. An overflow quantity or an underflow quantity is considered defined for purposes of this branch test.

In model 175, a jump out of the IWS voids the stack.

04ijK Branch to K if (Bi) = (Bj)

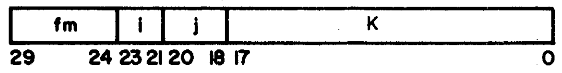


This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of the Bi and Bj registers. The branch to address K occurs only if the two quantities are identical on a bit-by-bit comparison basis. The current program sequence is continued for all other cases.

This instruction is for branching on an index equality test. A quantity consisting of all zeros and a quantity consisting of all ones are not equal for this test.

In model 175, a jump out of the IWS voids the stack.

05ijK Branch to K if (Bi) ≠ (Bj)

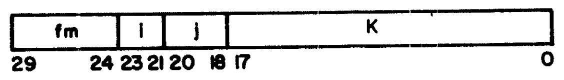


This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of the Bi and Bj registers. The program sequence is continued only if the two quantities are identical on a bit-by-bit comparison basis. The branch to address K occurs for all other cases.

This instruction is for branching on an index inequality test. A quantity consisting of all zeros and a quantity consisting of all ones are not equal for this test.

In model 175, a jump out of the IWS voids the stack.

06ijK Branch to K if (Bi) ≥ (Bj)

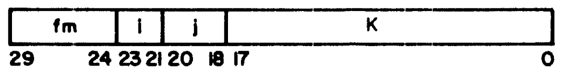


This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of the Bi and Bj registers. Both quantities are treated as signed integers. The branch to address K occurs if the content of Bi is greater than or equal to the content of Bj. The current program sequence is continued if the content of Bi is less than the content of Bj.

This instruction is for branching on an index threshold test. A positive zero quantity is considered greater than a negative zero quantity.

In model 175, a jump out of the IWS voids the stack.

07ijK Branch to K if (Bi) < (Bj)

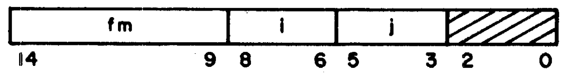


This instruction is a two-parcel instruction in which the lower-order 18 bits are used as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of the Bi and Bj registers. Both quantities are treated as signed integers. The branch to address K occurs if the content in Bi is less than the content in Bj. The current program sequence is continued if the content in Bi is greater than or equal to the content in Bj.

This instruction is for branching on an index threshold test. A positive zero quantity is considered greater than a negative zero quantity in this test.

In model 175, a jump out of the IWS voids the stack.

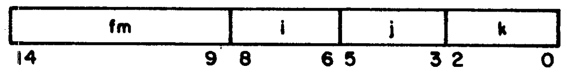
10ijx Transmit (Xj) to Xi



This instruction transfers a 60-bit word from the Xj register into the Xi register.

This instruction is for moving data from one X register to another X register. No logical function is performed on the data.

11ijk Logical Product of (Xj) and (Xk) to Xi

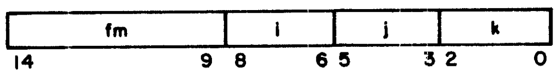


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The resultant word delivered to the Xi register is the bit-by-bit logical product of the two operands. Each of the 60 bits in Xj is compared with the corresponding bit in Xk to form a single bit in Xi. A sample computation is listed in octal notation to illustrate the operation performed and includes the four possible bit combinations that may occur.

```
(Xj) = 7777 7000 0123 4567 1010
(Xk) = 0123 4567 0077 7700 1100
(Xi) = 0123 4000 0023 4500 1000
```

This instruction is for extracting portions of a 60-bit word during data processing.

12ijk Logical Sum of (Xj) and (Xk) to Xi

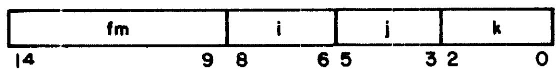


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The resultant word delivered to the Xi register is the bit-by-bit logical sum of the two operands. Each of the 60 bits in Xj is compared with the corresponding bit in Xk to form a single bit in Xi. A sample computation is listed in octal notation to illustrate the operation performed and includes the four possible bit combinations that may occur.

```
(Xj) = 0000 7777 0123 4567 1010
(Xk) = 0123 4567 7777 0000 1100
(Xi) = 0123 7777 7777 4567 1110
```

This instruction is for merging portions of a 60-bit word into a composite word during data processing.

13ijk Logical Difference of (Xj) and (Xk) to Xi

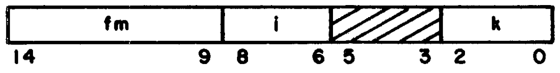


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The resultant word delivered to the Xi register is the bit-by-bit logical difference of the two operands. Each of the 60 bits in Xj is compared with the corresponding bit in Xk to form a single bit in Xi. A sample computation is listed in octal notation to illustrate the operation performed and includes the four possible bit combinations that may occur.

```
(Xj) = 0123 7777 0123 4567 1010
(Xk) = 0123 4567 7777 3210 1100
(Xi) = 0000 3210 7654 7777 0110
```

This instruction is for comparing bit patterns or for complementing bit patterns during data processing.

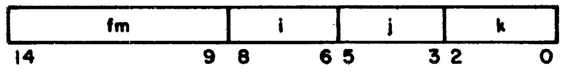
14ixk Transmit Complement of (Xk) to Xi



This instruction reads a 60-bit word from the Xk register, complements the word, and writes the result into the Xi register.

This instruction is for changing the sign of a fixed-point or floating-point quantity. This instruction is also useful for inverting an entire 60-bit field during data processing.

15ijk Logical Product of (Xj) and Complement of (Xk) to Xi

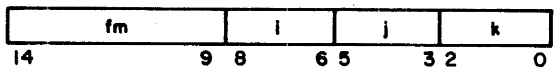


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The resultant word dèlivered to the Xi register is the bit-by-bit logical product of the value in Xj and the complement of the value in Xk. Each of the 60 bits in Xj is compared with the corresponding bit in Xk to form a single bit in Xi. A sample computation is listed in octal notation to illustrate the operation performed and includes the four possible bit combinations that may occur.

```
(Xj) = 7777 7000 0123 4567 1010
(Xk) = 0123 4567 0007 7700 1100
(Xi) = 7654 3000 0120 0067 0010
```

This instruction is for extracting portions of a 60-bit word during data processing.

16ijk Logical Sum of (Xj) and Complement of (Xk) to Xi

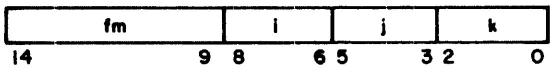


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The resultant word delivered to the Xi register is the bit-by-bit logical sum of the value in Xj and the complement of the value in Xk. Each of the 60 bits in Xj is compared with the corresponding bit in Xk to form a single bit in Xi. A sample computation is listed in octal notation to illustrate the operation performed and includes the four possible bit combinations that may occur.

```
(Xj) = 0000 7777 0123 4567 1010
(Xk) = 0123 4567 7777 0000 1100
(Xi) = 7654 7777 0123 7777 7677
```

This instruction is for merging portions of a 60-bit word into a composite word during data processing.

17ijk Logical Difference of (Xj) and Complement of (Xk) to Xi

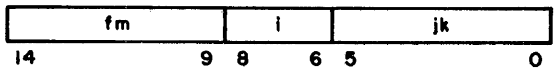


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The resultant word delivered to the Xi register is the bit-by-bit logical difference of the value in Xj and the complement of the value in Xk. Each of the 60 bits in Xj is compared with the corresponding bit in Xk to form a single bit in Xi. A sample computation is listed in octal notation to illustrate the operation performed and includes the four possible combinations that may occur.

```
(Xj) = 0123 7777 0123 4567 1010
(Xk) = 0123 4567 7777 3210 1100
(Xi) = 7777 4567 0123 0000 7667
```

This instruction is for comparing bit patterns or for complementing bit patterns during data processing.

20ijk Left Shift (Xi) by jk



This instruction reads one operand from the Xi register, shifts the 60-bit word left circularly by jk bit positions, and writes the resulting 60-bit word back into the same Xi register. The designators j and k are treated as a single six-bit positive integer operand in this instruction.

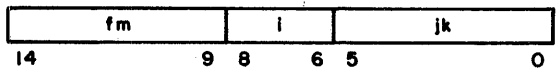
A left circular shift implies that the bit pattern in the 60-bit word is displaced towards the highest-order bit positions. The bits which are shifted off the upper end of the 60-bit word are inserted in the lowest-order bit positions in the same sequence. The resulting 60-bit word has the same quantity of bits with values of one and zero as in the original operand.

A sample computation is listed in octal notation to illustrate the operation performed.

```
Initial (Xi) = 2323 6600 0000 0000 0111
jk           = 12 (octal)
Final (Xi)   = 7540 0000 0000 0022 2464
```

This instruction, together with instruction 21, may be used whenever a data word is to be shifted by a predetermined amount. If the amount of shift is derived in the execution of the program, instruction 22 or 23 should be used.

21ijk Right Shift (Xi) by jk



This instruction reads one operand from the Xi register, shifts the 60-bit word right with sign extension by jk bit positions, and writes the resulting 60-bit word back into the same Xi register. The designators j and k are treated as a single six-bit positive integer operand in this instruction.

A right shift with sign extension implies that the bit pattern in the 60-bit word is displaced toward the lowest-order bit positions. The bits which are shifted off the lower end of the word are discarded. The highest-order bit positions are filled with copies of the original sign bit.

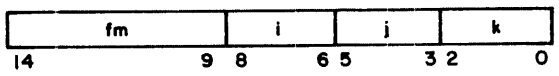
Two sample computations are listed in octal notation to illustrate the operation performed. The first example contains a positive operand and the second example contains a negative operand.

Initial (Xi) = 2004 7655 0002 3400 0004
jk = 30 (octal)
Final (Xi) = 0000 0000 2004 7655 0002

Initial (Xi) = 6000 4420 2222 0000 5643
jk = 10 (octal)
Final (Xi) = 7774 0011 0404 4440 0013

This instruction, together with instruction 20, may be used whenever a data word is to be shifted by a predetermined amount. If the amount of shift is derived in the execution of the program, instruction 22 or 23 should be used.

22ijk Left Shift (Xk) Nominally (Bj) Places to Xi



This instruction reads a 60-bit operand from the Xk register, shifts the data either left or right as specified by the content of Bj, and writes the resulting 60-bit word into the Xi register. If the value in Bj is positive, the data is shifted to the left in a circular mode the number of bit positions designated by the value in Bj. If the value in Bj is negative, the data is shifted to the right with sign extension the number of bit positions designated by the value in Bj. The sign of Bj is determined by Bj bit 17.

A left circular shift implies that the bit pattern in the 60-bit word is displaced towards the highest-order bit positions. The bits which are shifted off the upper end are inserted in the lowest-order bit positions in the same sequence. The resulting 60-bit word has the same quantity of bits with values of one and zero as in the original operand.

A right shift with sign extension implies that the bit pattern in the 60-bit word is displaced towards the lowest-order bit positions. The bits which are shifted off the lower end are discarded. The highest-order bit positions are filled with copies of the original sign bit.

Two sample computations are listed in octal notation to illustrate the operation performed. The first example contains a positive shift count resulting in a left circular shift. The second example illustrates the right shift with sign extension.

```
(Xk) = 2323 6600 0000 0000 0111
(Bj) = 00 0012
(Xi) = 7540 0000 0000 0022 2464

(Xk) = 1327 6000 0000 3333 2422
(Bj) = 77 7771
(Xi) = 0013 2760 0000 0033 3324
```

If Bj bits 6 through 10 are different than Bj bit 17, the shift count is greater than 63 (decimal), and a result of positive zero is returned. Bj bits 11 through 16 are not tested by this instruction.

This instruction is for use where the amount of shift is derived in the computation. This instruction is also useful for correcting the coefficient of a floating-point number when the exponent has been unpacked into a B register.

23ijk Right Shift (Xk) Nominally (Bj) Places to Xi

fm	i	j	k
14 9	8 6	5 3	2 0

This instruction reads a 60-bit operand from the Xk register, shifts the data either left or right as specified by the content of Bj, and writes the resulting 60-bit word into the Xi register. If the value in Bj is positive, the data is shifted to the right with sign extension the number of bit positions designated by the value in Bj. If the value in Bj is negative, the data is shifted to the left in a circular mode the number of bit positions designated by the value in Bj. The sign of Bj is determined by Bj bit 17.

A left circular shift implies that the bit pattern in the 60-bit word is displaced towards the highest-order bit positions. The bits which are shifted off the upper end are inserted in the lowest-order bit positions in the same sequence. The resulting 60-bit word has the same quantity of bits with values of one and zero as in the original operand.

A right shift with sign extension implies that the bit pattern in the 60-bit word is displaced towards the lowest-order bit positions. The bits which are shifted off the lower end of the word are discarded. The highest-order bit positions are filled with copies of the original sign bit.

Two sample computations are listed in octal notation to illustrate the operation performed. The first example contains a positive shift count and results in a right shift with sign extension. The second example contains a negative shift count and results in a left circular shift.

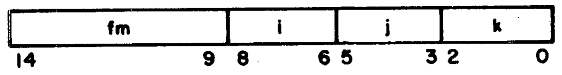
```
(Xk) = 1327 6000 0000 3333 2422
(Bj) = 00 0006
(Xi) = 0013 2760 0000 0033 3324

(Xk) = 2323 6600 0000 0000 0111
(Bj) = 77 7765
(Xi) = 7540 0000 0000 0022 2464
```

If Bj bits 6 through 10 are different than Bj bit 17, the shift count is greater than 63 (decimal), and a result of positive zero is returned. Bj bits 11 through 16 are not tested by this instruction.

This instruction is for use where the amount of shift is derived in the computation. This instruction is also useful for correcting the coefficient of a floating-point number when the exponent has been unpacked into a B register.

24ijk Normalize (Xk) to Xi and Bj



This instruction reads one operand from the Xk register, performs a normalizing operation on this word in a floating-point format, and delivers the normalized result to the Xi register. In addition, a positive integer shift count is sent to the Bj register. This shift count is the number of bit positions of shift required to normalize the original operand coefficient.

The normalizing operation consists of repositioning the coefficient portion of the operand and then adjusting the exponent portion of the operand to leave the value of the resulting word unaltered. The coefficient is shifted towards the higher-order bit positions of the word. The coefficient is shifted the minimum number of bit positions required to make bit 47 different from the sign bit 59. This places the most significant bit of the coefficient in the highest-order bit position. The exponent is then decreased by the number of bit positions shifted.

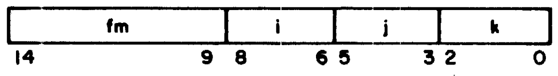
Two sample computations are listed in octal notation to illustrate the operation performed. The first example involves a positive floating-point number, and the second example involves a negative number.

```
(Xk) = 2034 0047 6500 0000 2262
(Xi) = 2026 4765 0000 0022 6200
(Bj) = 00 0006

(Xk) = 5743 7730 1277 7777 5515
(Xi) = 5751 3012 7777 7755 1577
(Bj) = 00 0006
```

Normalizing a number with either a positive or negative zero coefficient sets a shift count in Bj to 48 (decimal) and enters Xi with positive zero. If Xk contains an infinite quantity (3777xxx...x or 4000xxx...x) or an indefinite quantity (1777xxx...x or 6000xxx...x), no shift takes place. The content of Xk is copied to Xi, and Bj is set to zero. The corresponding infinite and indefinite exit conditions are also set in the CP for exit mode action.

25ijk Round Normalize (Xk) to Xi and Bj



This instruction reads one operand from the Xk register, performs a rounding and then a normalizing operation in floating-point format, and delivers the round normalized result to the Xi register. In addition, a positive integer shift count is sent to the Bj register. This shift count is the number of bit positions of shift required to normalize the original operand coefficient.

The rounding operation consists of adding a bit to the coefficient portion of the operand in a bit position immediately below the least significant bit position. This round bit has a value equal to the complement of the operand sign bit. The result is to increase the magnitude of the coefficient by one-half the value of the least significant bit.

The normalizing operation consists of repositioning the coefficient and adjusting the exponent to leave the value of the resulting floating-point quantity unaltered. The coefficient is shifted towards the higher-order bit positions. The round bit is shifted along with the coefficient. The displacement is the minimum number of bit positions required to make bit 47 different from the sign bit 59. This places the most significant bit of the coefficient in the highest-order bit position. The exponent is decreased by the number of bit positions shifted.

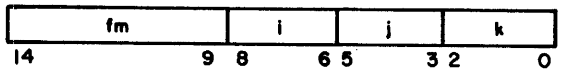
Two sample computations are listed in octal notation to illustrate the normalizing operation performed. The first example involves a positive floating-point number, and the second example involves a negative number.

(Xk) = 2034 0047 6500 0000 2262
(Xi) = 2026 4765 0000 0022 6240
(Bj) = 00 0006

(Xk) = 5743 7730 1277 7777 5515
(Xi) = 5751 3012 7777 7755 1537
(Bj) = 00 0006

If Xk contains either an infinite quantity (3777xxx...x or 4000xxx...x) or an indefinite quantity (1777xxx...x or 6000xxx...x), no shift takes place. The content of Xk is copied to Xi, and Bj is set to zero. The corresponding infinite and indefinite conditions are also set in the CP for exit mode action.

26ijk Unpack (Xk) to Xi and Bj



This instruction reads one operand from the Xk register, unpacks this word from floating-point format, and delivers the coefficient to the Xi register and the exponent to the Bj register. The 60-bit word delivered to the Xi register consists of the lowest 48 bits unaltered from the original operand plus the upper 12 bits, each equal to the original sign bit. This is a signed integer equal to the value of the coefficient in the original operand. The 18-bit quantity delivered to the Bj register is a signed integer equal to the value of the exponent in the original operand. The 11-bit exponent field in the operand is altered to remove the bias and then sign extended to fill out the 18-bit quantity. The sign of the coefficient is removed in this process.

Four sample sets of operands and unpacked results are listed in octal notation to illustrate the operation performed. These examples contain the four combinations of coefficient sign and exponent sign.

(Xk) = 2034 4500 3333 2000 0077
(Xi) = 0000 4500 3333 2000 0077
(Bj) = 00 0034

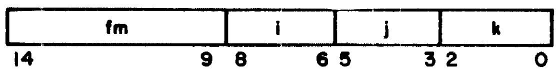
```
(Xk) = 1743 4500 3333 2000 0077
(Xi) = 0000 4500 3333 2000 0077
(Bj) = 77 7743

(Xk) = 5743 3277 4444 5777 7700
(Xi) = 7777 3277 4444 5777 7700
(Bj) = 00 0034

(Xk) = 6034 3277 4444 5777 7700
(Xi) = 7777 3277 4444 5777 7700
(Bj) = 77 7743
```

This instruction is for converting a number from floating-point format to fixed-point format.

27ijk Pack (Xk) and (Bj) to Xi



This instruction reads the contents of Xk and Bj, packs them into a single word in floating-point format, and delivers this result to the Xi register. The coefficient for the value in Xi is obtained from the content of Xk which is treated as a signed integer. The exponent for the value in Xi is obtained from the content of Bj which is treated as a signed integer.

The lowest-order 48 bits in Xi are copied directly from the lowest-order 48 bits in Xk. The sign bit in Xi is copied directly from the sign bit in Xk. The exponent field in Xi is derived from the value in Bj by extracting the lowest-order 11 bits in Bj and modifying this quantity for exponent bias and coefficient sign.

Four sample sets of operands and packed results are listed in octal notation to illustrate the operation performed. These examples contain the four combinations of coefficient sign and exponent sign.

```
(Xk) = 0000 4500 3333 2000 0077
(Bj) = 00 0034
(Xi) = 2034 4500 3333 2000 0077
```

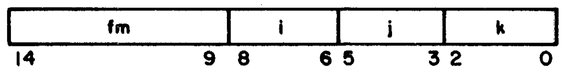
```
(Xk) = 0000 4500 3333 2000 0077
(Bj) = 77 7743
(Xi) = 1743 4500 3333 2000 0077

(Xk) = 7777 3277 4444 5777 7700
(Bj) = 00 0034
(Xi) = 5743 3277 4444 5777 7700

(Xk) = 7777 3277 4444 5777 7700
(Bj) = 77 7743
(Xi) = 6034 3277 4444 5777 7700
```

This instruction is for converting a number in fixed-point format to floating-point format.

30ijk Floating Sum of (Xj) and (Xk) to Xi



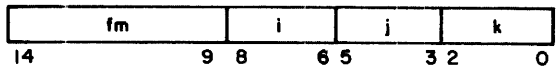
This instruction reads operands from two X registers, operates upon them to form a floating-point sum, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The sum of the quantities in Xj and Xk is delivered to the Xi register in floating-point format and is not necessarily normalized.

The two operands are unpacked from floating-point format, and the exponents are compared. The coefficient with the smallest exponent is right shifted by the difference of the exponents such that both coefficients are the same significance. The two coefficients are then added to form a 96-bit result. The upper half of the add result is then selected as a coefficient and packed along with the larger exponent to form the result sent to Xi. If coefficient overflow occurs, the sum is right shifted one place, and the exponent is increased by one.

If the two operands have unlike signs, the result coefficient may have leading zeros. No normalize operation is built into this instruction to correct this situation. A separate normalize instruction must be programmed if the result is to be kept in a normalized form.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

31ijk Floating Difference of (Xj) and (Xk) to Xi



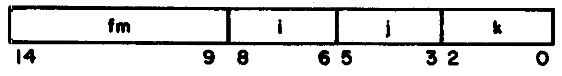
This instruction reads operands from two X registers, operates upon them to form a floating-point difference, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result of subtracting the quantity in Xk from the quantity in Xj is delivered to the Xi register in floating-point format and is not necessarily normalized.

The two operands are unpacked from floating-point format, and the exponents are compared. The coefficient with the smallest exponent is right shifted by the difference of the two exponents such that both coefficients are the same significance. The Xk coefficient is then subtracted from the Xj coefficient to form a 96-bit result. The upper half of the result is then selected and packed along with the larger exponent to form the result sent to Xi. If coefficient overflow occurs, the result is right shifted one place, and the exponent is increased by one.

If the two operands have like signs, the result coefficient may have leading zeros. No normalize operation is built into this instruction to correct this situation. A separate normalize instruction must be programmed if the result is to be kept in a normalized form.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

32ijk Floating Double-Precision Sum of (Xj) and (Xk) to Xi

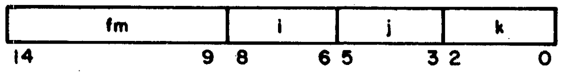


This instruction reads operands from two X registers, operates upon them to form a double-precision floating-point sum, and delivers the lower half of this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The sum of the quantities in Xj and Xk is delivered to the Xi register in floating-point format and is not necessarily normalized.

The two operands are unpacked from floating-point format, and the exponents are compared. The coefficient with the smallest exponent is right shifted by the difference of the two exponents such that both coefficients are the same significance. The coefficients are then added to form a 96-bit result. The lower half of the result is then selected and packed along with the larger exponent minus 48 (decimal) to form the result sent to Xi. If coefficient overflow occurs, the result is right shifted by one place, and the exponent is increased by one.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

33ijk Floating Double-Precision Difference of (Xj) and (Xk) to Xi

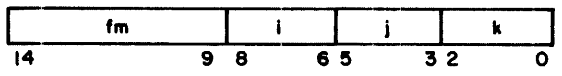


This instruction reads operands from two X registers, operates upon them to form a double-precision floating-point difference, and delivers the lower half of this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result of subtracting the quantity in Xk from the quantity in Xj is delivered to the Xi register in floating-point format and is not necessarily normalized.

The two operands are unpacked from floating-point format, and the exponents are compared. The coefficient with the smallest exponent is right shifted by the difference of two exponents such that both coefficients are the same significance. The Xk coefficient is then subtracted from the Xj coefficient to form a 96-bit result. The lower half of the result is then selected and packed along with the largest exponent minus 48 (decimal) to form the result sent to Xi. If coefficient overflow occurs, the result is right shifted one place, and the exponent is increased by one.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

34ijk Round Floating Sum of (Xj) and (Xk) to Xi

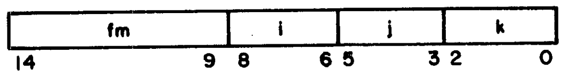


This instruction reads operands from two X registers, operates upon them to form a rounded floating-point sum, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result is delivered to the Xi register in floating-point format and is not necessarily normalized.

The round floating-point sum is a single-precision floating-point sum with a round bit (or bits) inserted before the add operation takes place. The round bit is inserted after one of the coefficients has been shifted by the difference of the exponents. A round bit is always inserted in the coefficient with the larger exponent. If the exponents are equal, the round bit is inserted in the coefficient for Xk. The round bit is equal to the complement of the sign bit and is inserted immediately to the right of the lowest-order bit in the coefficient. This has the effect of increasing the magnitude of the coefficient by one-half of the least significant bit. A second round bit is inserted in a corresponding manner to the other coefficient if both operands have unlike signs.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

35ijk Round Floating Difference of (Xj) and (Xk) to Xi



This instruction reads operands from two X registers, operates upon them to form a rounded floating-point difference, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result of subtracting the quantity in Xk from the quantity in Xj is delivered to the Xi register in floating-point format and is not necessarily normalized.

The round floating-point difference is a single-precision floating-point difference with a round bit (or bits) inserted before the subtract operation takes place. The round bit is inserted after one of the coefficients has been shifted by the difference of the two exponents. A round bit is always inserted in the coefficient with the larger exponent. If the exponents are equal, the round bit is added to the coefficient for Xk. The round bit is equal to the complement of the sign bit and is inserted immediately to the right of the lowest-order bit in the coefficient. This has the effect of increasing the magnitude of the coefficient by one-half of the least significant bit. A second round bit is inserted in a corresponding manner to the other coefficient if both operands are normalized or if the two operands have unlike signs.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

36ijk Integer Sum of (Xj) and (Xk) to Xi

fm	i	j	k
14 9	8 6	5 3	2 0

This instruction reads operands from two X registers, operates upon them to form a 60-bit integer sum, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are signed integers. The resulting integer sum is delivered to the Xi register. Overflow is not detected.

This instruction is for addition of integers too large for handling by 50 through 77 instructions. This instruction is also useful in merging and comparing data fields during data processing.

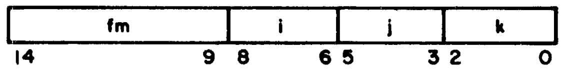
37ijk Integer Difference of (Xj) and (Xk) to Xi

fm	i	j	k
14 9	8 6	5 3	2 0

This instruction reads operands from two X registers, operates upon them to form a 60-bit integer difference, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are signed integers. The result of subtracting the quantity in Xk from the quantity in Xj is delivered to the Xi register. Overflow is not detected.

This instruction is for subtraction of integers too large for handling by 50 through 77 instructions. This instruction is also useful in comparing data fields during data processing.

40ijk Floating Product of (Xj) and (Xk) to Xi



This instruction reads operands from two X registers, operates upon them to form a floating-point product, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result is delivered to the Xi register in floating-point format. If both operands are normalized, the result is also normalized. If both operands are not normalized, the result is not normalized.

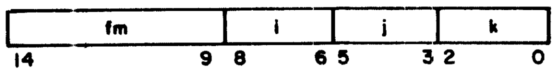
The two operands are unpacked from floating-point format. The exponents are added with a correction factor to determine the exponent for the result. The coefficients are multiplied as signed integers to form a 96-bit integer product. The upper half of this product is extracted to form the coefficient for the result. If the original operands are normalized and the product has only 95 significant bits, a one-bit left shift to normalize the result coefficient is done. The resulting exponent is reduced by one count in this case.

If the two operands are not both normalized, the resulting double-precision product has less than 96 significant bits. No test is made for the position of the most significant bit. The upper 48 bits are read from the double-precision product register. Leading zeros occur in this result coefficient.

This instruction is for use in floating-point calculations where rounding of operands is not desired, such as in multiple-precision arithmetic and in calculations involving error analysis.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

41ijk Round Floating Product of (Xj) and (Xk) to Xi



This instruction reads operands from two X registers, operates upon them to form a rounded floating-point product, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result is delivered to the Xi register in floating-point format. If both operands are normalized, the result is also normalized. If both operands are not normalized, the result is not normalized.

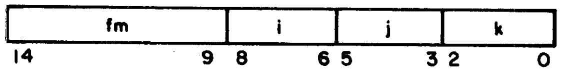
The two operands are unpacked from floating-point format. The exponents are added with a correction factor to determine the exponent for the result. The coefficients are multiplied as signed integers to form a 96-bit integer product. A rounding bit is added to bit position 46 of this product. The upper half of this product is extracted to form the coefficient for the result. If the original operands are normalized and the product has only 95 significant bits, a one-bit left shift to normalize the result coefficient is done. The resulting exponent is reduced by one count in this case.

If the two operands are not both normalized, the resulting double-precision product has less than 96 significant bits. No test is made for the position of the most significant bit. The upper 48 bits are read from the double-precision product register. Leading zeros occur in this result coefficient.

This instruction is for use in single-precision floating-point calculations. For multiple-precision calculations, the 40 and 42 instructions must be used.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

42ijk Floating Double-Precision Product of (Xj) and (Xk) to Xi



This instruction reads operands from two X registers, operates upon them to form a double-precision floating-point product, and delivers the lower half of this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The lower half of the double-precision product is delivered to the Xi register in floating-point format and is not necessarily normalized.

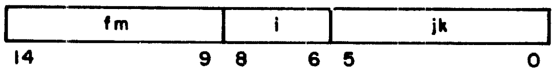
The operands are not rounded in this operation. The two operands are unpacked from floating-point format. The exponents are added to determine the exponent for the result. The result exponent is exactly 48 less than the exponent for a 40 instruction. The coefficients are multiplied as signed integers to form a 96-bit integer product. The lower half of this product is extracted to form the coefficient for the result. If the original operands are normalized and the double-precision product has only 95 significant bits, a one-bit left shift to normalize the result coefficient is done. The resulting exponent is reduced by one count in this case.

If both operands are not normalized, the resulting double-precision product has less than 96 significant bits. No test is made for the position of the most significant bit in the product. The lower 48 bits are always read from the 96-bit product register.

This instruction is for use in multiple-precision floating-point calculations. This instruction also provides for integer multiplication capabilities where both operands have an exponent value of plus or minus zero and neither coefficient has been normalized. The integer result sent to the Xi register is 48 bits with 60-bit sign extension. If the result exceeds 48 bits, the hardware does not detect an overflow. An overflow check can be made by executing a 40 instruction using the same two operands. Then, if the result is nonzero, overflow is indicated. An integer multiply operation is not intended to be used with normalized operands.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action. (Refer to Central Processor Differences in section 5.)

43ijk Form Mask of jk Bits to Xi

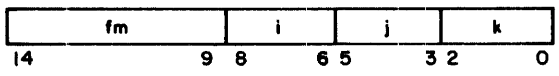


This instruction generates a masking word using the j and k designators as parameters. No operands are read from operating registers. The j and k designators are treated as a single six-bit quantity to designate the width of the masking field. A field of ones, beginning at the highest-order end of the word, is extended downward on a background of zeros. The completed masking word consists of one bits in the highest-order jk bit positions and zero bits in the remainder of the word. This masking word is then delivered to the Xi register. The following are sample parameters.

j	=	2
k	=	4
Xi	=	7777 7760 0000 0000 0000

This instruction is for generating variable width masks for logical operations. This instruction, together with a shift instruction, generally creates an arbitrary field mask faster than reading a pregenerated mask from CM.

44ijk Floating Divide (Xj) by (Xk) to Xi



This instruction reads operands from two X registers, operates upon them to form a floating-point quotient, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format. The result of dividing the content of Xj by the content of Xk is delivered to the Xi register. If both operands are normalized, the quotient is also normalized. The remainder from the division process is discarded.

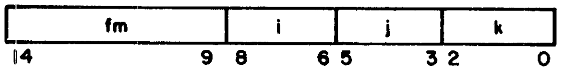
The two operands are unpacked from floating-point format. The exponents are subtracted with a correction factor to determine the exponent for the result. The coefficient from Xj is positioned in a dividend register. The coefficient from Xk is trial-subtracted repeatedly from the dividend. The quotient bits are assembled in a quotient register. When 48 bits of the quotient have been assembled, they are packed with the result exponent into floating-point format and delivered to the Xi register.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action.

If the dividend is not normalized, the quotient cannot be normalized. However, the quotient is correct even though there may be leading zeros in the coefficient. If the divisor is not normalized, the quotient may be incorrect. If the coefficient for the value in Xj is larger than the coefficient for the value in Xk by a factor of two or more, a divide fault causes an indefinite result to be returned to Xi. (Refer to Floating-Point Arithmetic in section 5.)

This instruction is for use in floating-point calculations where rounding of operands is not desired. In multiple-precision division, this instruction must be followed by a multiplication of the quotient by the divisor and subtracted from the dividend to reconstruct the remainder.

45ijk Round Floating Divide (Xj) by (Xk) to Xi



This instruction reads operands from two X registers, operates upon them to form a rounded floating-point quotient, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format. The result of dividing the content of Xj by the content of Xk is delivered to the Xi register. If both operands are normalized, the quotient is also normalized. The remainder from the division process is discarded.

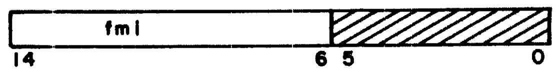
The two operands are unpacked from floating-point format in this operation. The exponents are subtracted with a correction factor to determine the exponent for the result. The coefficient from Xj is positioned in a dividend register. The Xj quantity is modified by inserting a 2525...25 round pattern below the lowest-order bit of the dividend coefficient. The coefficient from Xk is trial-subtracted repeatedly from the dividend. The quotient bits are assembled in a quotient register. When 48 bits of the quotient have been assembled, they are packed with the result exponent into floating-point format and delivered to the Xi register.

If the dividend is not normalized, the quotient cannot be normalized. However, the quotient is correct even though there may be leading zeros in the coefficient. If the divisor is not normalized, the quotient may be incorrect. If the coefficient for the value in Xj is larger than the coefficient for the value in Xk by a factor of two or more, a divide fault occurs. A divide fault causes an indefinite result to be returned to Xi. (Refer to Floating-Point Arithmetic in section 5.)

This instruction is for use in single-precision floating-point calculations where rounding of operands is desired to reduce truncation errors.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, corresponding exit conditions are set in the CP for exit mode action.

460xx through 463xx Pass

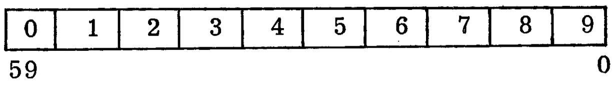


These instructions are used to fill program instruction words where necessary to match jump destinations with word boundaries. The j and k designators are ignored, and a nonzero value has no effect in this instruction.

464 Through 467 Compare/Move Instructions (Models 172 Through 174)

These instructions must appear in parcel 0 or be treated as illegal instructions. For model 175, these instructions are illegal. (Refer to Illegal Instructions in section 5.)

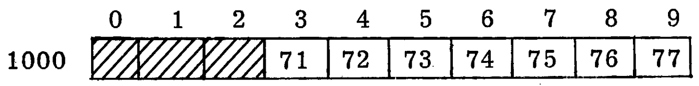
Data fields consisting of 6-bit characters may start or end with any character position (offset) of the 10 6-bit positions in each word. The character positions are designated as follows:



For move instructions, the designator K1 specifies which CM word contains the first character of the source data field, and designator C1 specifies the character position (offset) of the first character. Designator K2 specifies the CM location in which the first character of the result data field is placed, and designator C2 specifies the first character position. For compare instructions, both data field addresses specify source fields.

Example:

If the instruction is K1=1000 and C1=3, the first character of the source field is in position 3 of location 1000.



Therefore, the first character of the source field is 71.

An address is out-of-range if C1 or C2 is greater than 9_{10}, K1 plus N1 is greater than the relative address plus the program field length (RAC+FLC) (N1 equals number of CM references made to the source data field starting at K1), or K2 plus N2 is greater than (RAC+FLC) (N2 equals number of CM references made to the result data field starting at K2). The address out-of-range condition is not predicted. When the condition occurs, some unpredictable part of the operation is performed. The amount of the operation performed does not necessarily repeat on an identical out-of-range condition.

LL is the lower four bits and LU is the upper nine bits of the field length designator in numbers of characters. The maximum length of the data fields for the move direct and the compare instructions is 177_8 (127_{10}) characters. The maximum data field length for the move indirect instruction is 17777_8 (8191_{10}) characters. If L (LU and LL combined) is zero, the instruction becomes a pass.

For overlapping move instructions, the address of the source field (specified by K1) must be greater than the address of the result field (specified by K2) to provide proper field overlap. If K1 is less than K2, part of the source field is changed during execution, with the amount of change determined by the number of CM conflicts encountered. Overlapping fields should not contain more than 377_8 characters, because an exchange jump interrupts any compare/move operation having a decremented field length greater than 377_8.

464jK Move Indirect (Models 172 through 174)

This instruction moves the source field to the result field as specified by the 60-bit descriptor word (Figure 4-2). The quantity in Bj plus K is the address in CM of the descriptor word. The move is from left to right through the field. The X0 register clears at the end of execution. Any instructions located in the lower two parcels of the instruction word are not executed.

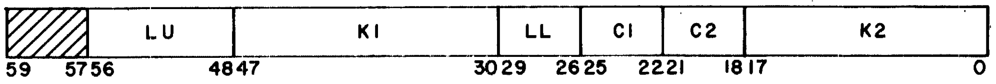


Figure 4-2. Descriptor Word

465 Move Direct (Models 172 through 174)

This instruction moves the source field to the result field as specified by the 60-bit instruction word (Figure 4-3). The field length is limited to a seven-bit count.

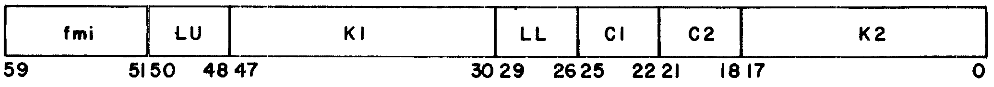


Figure 4-3. Compare/Move Instruction Format

466 Compare Collated (Models 172 through 174)

This instruction compares the field designated by K1, C1 with the field designated by K2, C2 as specified by the 60-bit instruction word (Figure 4-3). The X0 register is then set prior to instruction termination as follows:

If field K1 is greater than field K2, set X0 to 0000 0000 0000 0000 0xxx

If field K1 is equal to field K2, set X0 to 0000 0000 0000 0000 0000

If field K1 is less than field K2, set X0 to 7777 7777 7777 7777 7yyy where yyy is the complement of xxx.

The compare is from left to right through the fields until two unequal characters are found. These two characters are then collated, looked up in the collating table (Table 4-2) beginning at address A0. If the table values found for the two unequal characters are the same, the compare continues until another pair of characters is unequal or until the field length is exhausted. If the table values found for the two unequal characters are unequal, X0 is set according to the preceding rules.

The value of the three octal numbers xxx, stored in X0, is determined by the equation L minus N equals xxx, where L is the length of the field and N is the number of pairs of characters that were collated equal, prior to instruction termination. In other words, xxx is the number of pairs of characters not yet compared plus one.

The A0 register contains the starting word address of an 8-word, 64-character collating table (Table 4-2). This table must have been previously stored in consecutive CM locations.

The collated value of a character is found by examining the collating table. The upper three bits of the character to be collated are added to A0 to obtain the relative address of the word containing the collated value. The lower three bits of the character to be collated specify the character address of the collated value.

Example:

Suppose the character under examination is an octal 63. The 6 is added to the A0 to form the word address. The 3 is used to pick the correct character from that word. The value of 63 is 63 in the collating table.

TABLE 4-2. COLLATING TABLE

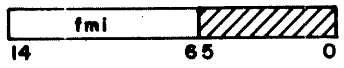
Address	Collating Character Locations									
A0	00	01	02	03	04	05	06	07	xx	xx
A0+1	10	11	12	13	14	15	16	17	xx	xx
A0+2	20	21	22	23	24	25	26	27	xx	xx
A0+3	30	31	32	33	34	35	36	37	xx	xx
A0+4	40	41	42	43	44	45	46	47	xx	xx
A0+5	50	51	52	53	54	55	56	57	xx	xx
A0+6	60	61	62	63	64	65	66	67	xx	xx
A0+7	70	71	72	73	74	75	76	77	xx	xx

59 0

467 Compare Uncollated (Models 172 through 174)

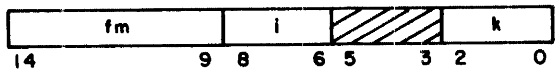
This instruction is similar to the 466 instruction except that the collating table is not used. The X0 register is set when the first pair of unequal characters is encountered or when the field length is exhausted.

464 through 467 Instructions (Model 175)



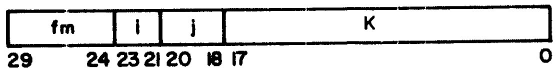
These instructions are illegal instructions. (Refer to Illegal Instructions in section 5.)

47ixk Population Count of (Xk) to Xi



This instruction reads one operand from the Xk register, counts the number of one bits in the operand, and stores the count in the Xi register. The count delivered to the Xi register is a positive integer. If the operand is all ones, a count of 60 (decimal) is delivered to the Xi register. If the operand is all zeros, a zero word is delivered to the Xi register.

50ijK Set Ai to (Aj) + K

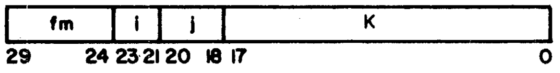


This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Aj register, forms the sum of the operand plus K, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

This instruction is for fetching operands from CM for computation and for delivering results back into CM.

51ijK Set Ai to (Bj) + K

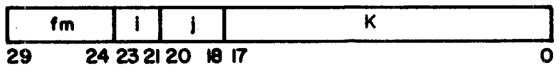


This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Bj register, forms the sum of the operand plus K, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

This instruction is for fetching operands from CM for computation and for delivering results back into CM.

52ijK Set Ai to (Xj) + K

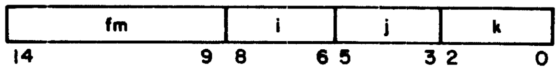


This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Xj register, forms the sum of the operand plus K, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

This instruction is for fetching operands from CM for computation and for delivering results back into CM.

53ijk Set Ai to (Xj) + (Bk)

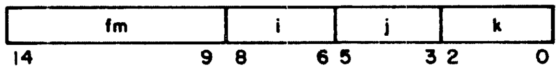


This instruction reads operands from the Xj and Bk registers, forms the sum of the operands, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

The instruction is for fetching operands from CM for computation and for delivering results back into CM.

54ijk Set Ai to (Aj) + (Bk)

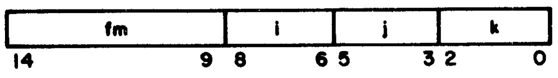


This instruction reads operands from the Aj and Bk registers, forms the sum of the operands, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

This instruction is for fetching operands from CM for computation and for delivering results back into CM.

55ijk Set Ai to (Aj) - (Bk)

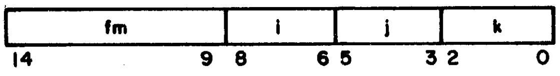


This instruction reads operands from the Aj and Bk registers, subtracts the Bk operand from the Aj operand, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

This instruction is for fetching operands from CM for computation and for delivering results back into CM.

56ijk Set Ai to (Bj) + (Bk)

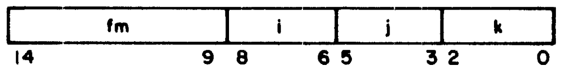


This instruction reads operands from the Bj and Bk registers, forms the sum of the operands, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

This instruction is for fetching operands from CM for computation and for delivering results back into CM.

57ijk Set Ai to (Bj) - (Bk)

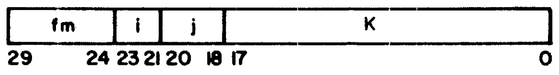


This instruction reads operands from the Bj and Bk registers, subtracts the Bk operand from the Bj operand, and delivers the result to the Ai register. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

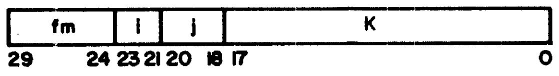
This instruction is for fetching operands from CM for computation and for delivering results back into CM.

60ijK Set Bi to (Aj) + K



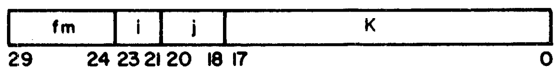
This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Aj register, forms the sum of the operand plus K, and delivers the result to the Bi register. The sum is formed in an 18-bit ones-complement mode. This instruction is for address modification in the increment registers.

61ijK Set Bi to (Bj) + K



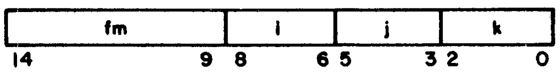
This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Bj register, forms the sum of the operand plus K, and delivers this result to the Bi register. The sum is formed in an 18-bit ones-complement mode.

62ijK Set Bi to (Xj) + K



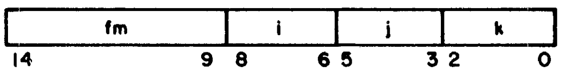
This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Xj register, forms the sum of the operand plus K, and delivers the result to the Bi register. The sum is formed in an 18-bit ones-complement mode.

63ijk Set Bi to (Xj) + (Bk)



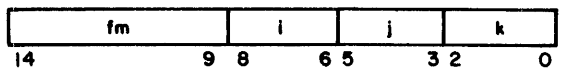
This instruction reads operands from the Xj and Bk registers, adds the operands, and delivers the result to the Bi register. The sum is formed in an 18-bit ones-complement mode.

64ijk Set Bi to (Aj) + (Bk)



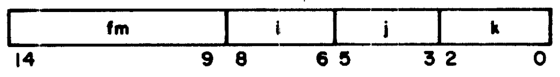
This instruction reads operands from the Aj and Bk registers, adds the operands, and delivers the result to the Bi register. The sum is formed in an 18-bit ones-complement mode.

65ijk Set Bi to (Aj) - (Bk)



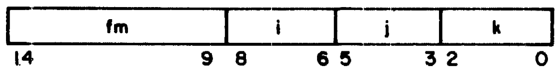
This instruction reads operands from the Aj and Bk registers, subtracts the Bk operand from the Aj operand, and delivers the result to the Bi register. The difference is formed in an 18-bit ones-complement mode.

66ijk Set Bi to (Bj) + (Bk)



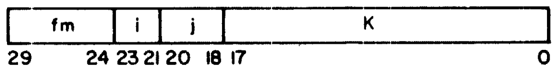
This instruction reads operands from the Bj and Bk registers, adds the operands, and delivers the result to the Bi register. The sum is formed in an 18-bit ones-complement mode.

67ijk Set Bi to (Bj) - (Bk)



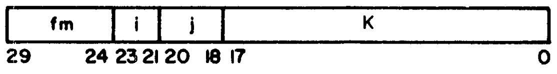
This instruction reads operands from the Bj and Bk registers, subtracts the Bk operand from the Bj operand, and delivers the result to the Bi register. The difference is formed in an 18-bit ones-complement mode.

70ijK Set Xi to (Aj) + K



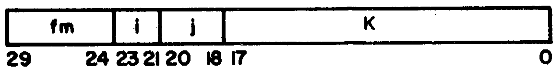
This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Aj register, forms the sum of the operand plus K, and delivers the result to the Xi register. The sum is formed in an 18-bit ones complement mode. The resulting 18-bit quantity is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

71ijK Set Xi to (Bj) + K



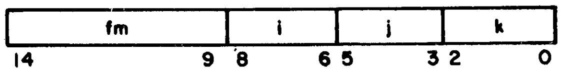
This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Bj register, forms the sum of the operand plus K, and delivers the result to the Xi register. The sum is formed in an 18-bit ones-complement mode. The resulting 18-bit quantity is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

72ijK Set Xi to (Xj) + K



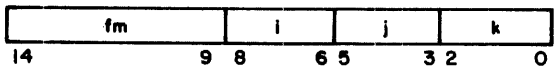
This is a two-parcel instruction in which the lower-order 18 bits are used as operand K. This instruction reads an operand from the Xj register, forms the sum of the operand plus K, and delivers the result to the Xi register. The sum is formed in an 18-bit ones-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

73ijk R Set Xi to (Xj) + (Bk)



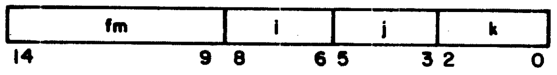
This instruction reads operands from the Xj and Bk registers, adds the operands, and delivers the result to the Xi register. The sum is formed in an 18-bit ones-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

74ijk Set Xi to (Aj) + (Bk)



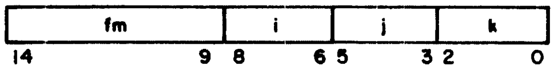
This instruction reads operands from the Aj and Bk registers, adds the operands, and delivers the result to the Xi register. The sum is formed in an 18-bit ones-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

75ijk Set Xi to (Aj) - (Bk)



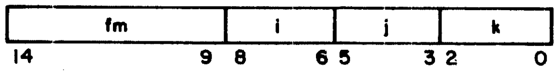
This instruction reads operands from the Aj and Bk registers, subtracts the Bk operand from the Aj operand, and delivers the result to the Xi register. The difference is formed in an 18-bit ones-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

76ijk Set Xi to (Bj) + (Bk)



This instruction reads operands from the Bj and Bk registers, adds the operands, and delivers the result to the Xi register. The sum is formed in an 18-bit ones-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

77ijk Set Xi to (Bj) - (Bk)



This instruction reads operands from the Bj and Bk registers, subtracts the Bk operand from the Bj operand, and delivers the result to the Xi register. The difference is formed in an 18-bit ones-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in the Xi register.

PERIPHERAL PROCESSOR INSTRUCTIONS

This part of the manual describes the PP instructions. Each instruction is described separately.

INSTRUCTION FORMATS

Two instruction formats are used. The 12-bit format (Figure 4-4) has a 6-bit operation code f and a 6-bit operand or operand address d. The 24-bit format (Figure 4-5) uses the 12-bit quantity m, the content of the next program address (P plus 1), with d to form an 18-bit operand or 12-bit operand address.

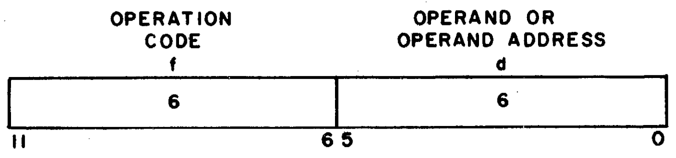


Figure 4-4. 12-Bit Instruction Format

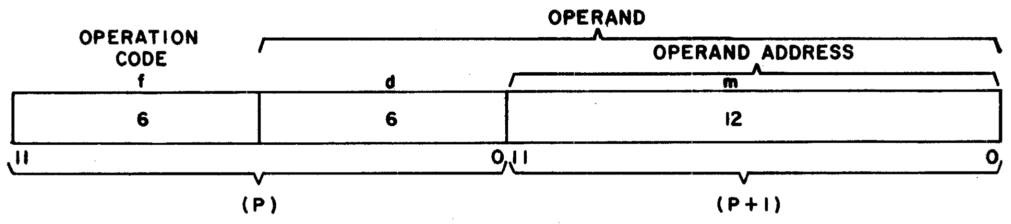


Figure 4-5. 24-Bit Instruction Format

INSTRUCTION DESIGNATOR DESCRIPTIONS

Instruction designators are listed and described in Table 4-3.

TABLE 4-3. PERIPHERAL PROCESSOR INSTRUCTION DESIGNATORS

Designator	Use
A	A register
d	6-bit operand address, jump count, shift count, channel designator, or additional instruction code
f	12-bit operand address, 12-bit jump address, 12-bit function code, or 12-bit quantity used with d to form an 18-bit operand
m	12-bit quantity used with d to form an 18-bit operand or operand address
P	Program address register
Q	Q register
( )	Content of a register or location
(( ))	Indirect addressing

INSTRUCTION ADDRESSING MODES

Several addressing modes permit PP program indexing and the manipulation of operands. The addressing modes consist of no address, direct address, and indirect address. These modes are summarized in Table 4-4.

No Address

In this mode, d or dm is taken directly as an operand. This mode eliminates the need for storing constants. The d quantity is considered as a 12-bit number, the upper 6 bits of which are zero. The dm quantity has d as the upper 6 bits and m as the lower 12 bits.

Direct Address

In this mode, d or m plus the value in d is used as the address of the operand. The d quantity specifies one of the first 64 addresses in PPM (0000 through 0077_8). m plus the value in d generates a 12-bit address for referencing all possible PPM locations (0000 through 7777_8). If d is not 0, the content of address d is added to m to produce an operand address (indexed addressing). If d is 0, m is taken as the operand address (direct addressing).

Indirect Address

In this mode, d specifies an address in which the content is the address of the desired operand. Thus, d specifies the operand address indirectly. Indirect addressing and indexed addressing require an additional PPM reference over direct addressing.

TABLE 4-4. ADDRESSING MODES FOR PERIPHERAL PROCESSOR INSTRUCTIONS

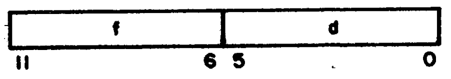
Instruction Types	Addressing Mode		
	Direct	Indirect	No Address
Load	30, 50	40	14, 20
Add	31, 51	41	16, 21
Subtract	32, 52	42	17
Logical difference	33, 53	43	11, 23
Store	34, 54	44	-
Replace add	35, 55	45	-
Replace add one	36, 56	46	-
Replace subtract one	37, 57	47	-
Long jump	01	-	-
Return jump	02	-	-
Unconditional jump	-	-	03
Zero jump	-	-	04
Nonzero jump	-	-	05
Positive jump	-	-	06
Minus jump	-	-	07
Shift	-	-	10
Logical product	-	-	12, 22
Selective clear	-	-	13
Load complement	-	-	15

INSTRUCTION DESCRIPTIONS

The following PP instructions are described separately. Shaded areas, like those in the 260x and 261x instruction formats, indicate unused bits. The unused bits are ignored by the PPs.

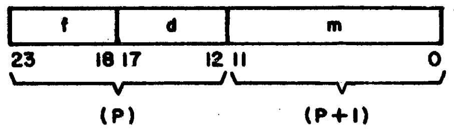
Timing information for the PP instructions is in appendix A.

0000 Pass



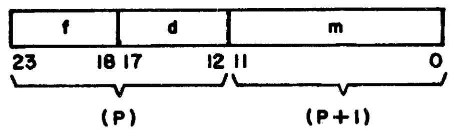
This 12-bit instruction specifies that no operation is to be performed. The instruction provides a means of padding out a program.

01dm Long Jump to m + (d)



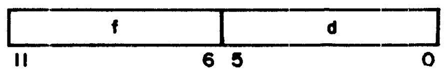
This 24-bit instruction jumps to the address given by m plus the content of location d. If d equals zero, m is not modified.

02dm Return Jump to m + (d)



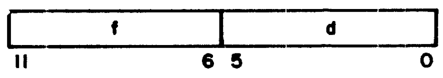
This 24-bit instruction jumps to the address given by m plus the content of location d. If d equals zero, m is not modified. The current program address (P) plus 2 is stored at the jump address. The next instruction starts at the jump address plus 1. The subprogram should exit with a long jump, or normal sequencing to the jump address minus 1, which should in turn contain a long jump, 0100. This returns the original program address plus 2 to the P register.

03d Unconditional Jump d



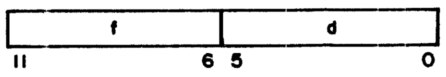
This 12-bit instruction provides an unconditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. The value of d is added to the current program address. If d is positive (01 through 37), 0001 through 0037 is added, and the jump is forward. If d is negative (40 through 76), 7740 through 7776 is added, and the jump is backward. The program hangs when d equals 00 or 77 and requires a deadstart to restart the system.

04d Zero Jump d



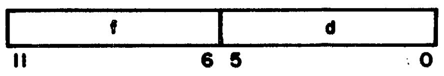
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the content of the A register is zero, the jump is taken. If the content of A is nonzero, the next instruction is executed from P plus 1. Negative zero (777777) is treated as nonzero. For interpretation of d, refer to instruction 03.

05d Nonzero Jump d



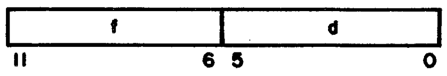
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the content of the A register is nonzero, the jump is taken. If the content of A is zero, the next instruction is executed from P plus 1. Negative zero (777777) is treated as nonzero. For interpretation of d, refer to instruction 03.

06d Plus Jump d



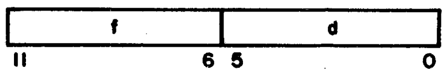
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the sign of the A register is positive, the jump is taken. If the sign of A is negative, the next instruction is executed from P plus 1. Positive zero is treated as a positive quantity. Negative zero is treated as a negative quantity. For interpretation of d, refer to instruction 03.

07d Minus Jump d



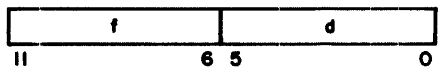
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the content of the A register is negative, the jump is taken. If the content of A is positive, the next instruction is executed from P plus 1. Positive zero is treated as a positive quantity. Negative zero is treated as a negative quantity. For interpretation of d, refer to instruction 03.

10d Shift d



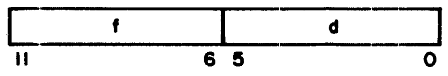
This 12-bit instruction shifts the content of the A register right or left d places. If d is positive (00 through 37), the shift is left circular. If d is negative (40 through 77), the shift is right (end-off with no sign extension). Thus, d equal to 06 requires a left shift of six places; d equal to 71 requires a right shift of six places.

11d Logical Difference d



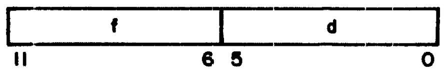
This 12-bit instruction forms in the A register the bit-by-bit logical difference of d and the lower 6 bits of A. This is equivalent to complementing individual bits of A that correspond to bits of d that are one. The upper 12 bits of A are not altered.

12d Logical Product d



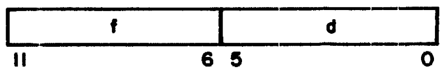
This 12-bit instruction forms the bit-by-bit logical product of d and the lower 6 bits of the A register, and leaves this quantity in the lower 6 bits of A. The upper 12 bits of A are zero.

13d Selective Clear d



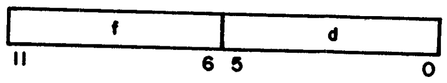
This 12-bit instruction clears any of the lower 6 bits of the A register where corresponding bits of d are one. The upper 12 bits of A are not altered.

14d Load d



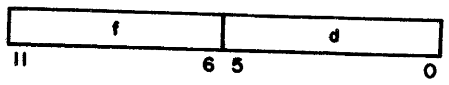
This 12-bit instruction clears the A register and loads d. The upper 12 bits of A are zero.

15d Load Complement d



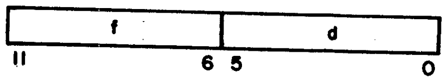
This 12-bit instruction clears the A register and loads the complement of d. The upper 12 bits of A are one.

16d Add d



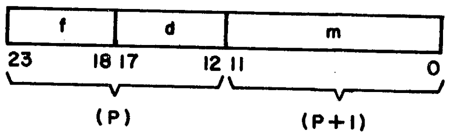
This 12-bit instruction adds d (treated as a 6-bit positive quantity) to the content of the A register

17d Subtract d



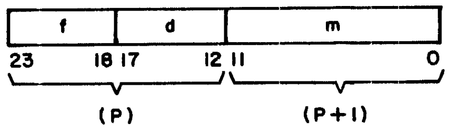
This 12-bit instruction subtracts d (treated as a 6-bit positive quantity) from the content of the A register.

20dm Load dm



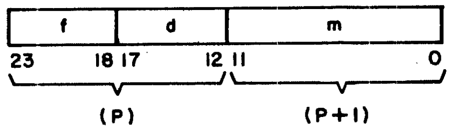
This 24-bit instruction clears the A register and loads an 18-bit quantity consisting of d as the upper 6 bits and m as the lower 12 bits. The content of the location following the present program address is read to provide m.

21dm Add dm



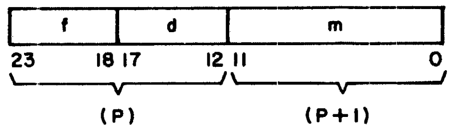
This 24-bit instruction adds to the A register the 18-bit quantity consisting of d as the upper 6 bits and m as the lower 12 bits. The content of the location following the present program address (P plus 1) is read to provide m.

22dm Logical Product dm



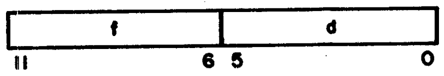
This 24-bit instruction forms in the A register the bit-by-bit logical product of the content of the A register and the 18-bit quantity dm. The upper 6 bits of this quantity consist of d, and the lower 12 bits are the content of the location following the present program address (P plus 1).

23dm Logical Difference dm



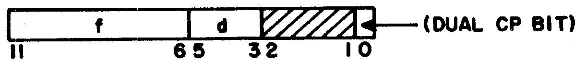
This 24-bit instruction forms in the A register the bit-by-bit logical difference of the content of the A register and the 18-bit quantity dm. This is equivalent to complementing individual bits of A which correspond to bits of dm that are one. The upper 6 bits of the quantity consist of d, and the lower 12 bits are the content of the location following the present program address (P plus 1).

2400, 2500 Pass



These 12-bit instructions specify that no operation is to be performed. These instructions provide a means of padding out a program.

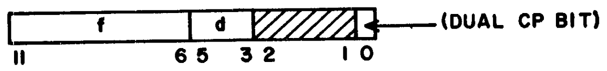
260x Exchange Jump



The 12-bit instruction transmits an 18-bit, absolute, address from the A register to the CP with a signal which tells the CP to perform an exchange jump. The address in A is the starting location of an exchange package of 16 words containing information for a CP program to be executed. The 18-bit initial address must be entered in A before this instruction is executed. The CP replaces the exchange package with an exchange package from the interrupted CP program. The PP is not interrupted.

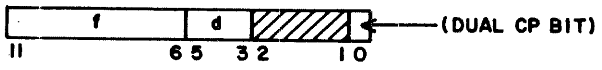
In model 174, the lowest-order bit of the instruction format specifies which of the two CPs the exchange jump interrupts. In models 172, 173, and 175, this bit is not interpreted.

261x Monitor Exchange Jump



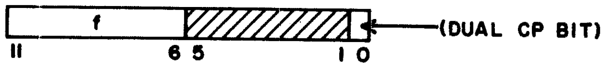
This 12-bit instruction is enabled or disabled by the CEJ/MEJ switch. When the switch is in the ENABLE position, the instruction causes a conditional exchange jump of the CP. If the monitor flag clears, this instruction initiates the exchange jump and sets the flag. If the monitor flag sets, this instruction acts as a pass instruction. The starting address for this exchange is the 18-bit address held in the PP A register. (The PP program must have loaded A with an appropriate address prior to executing this instruction.) This starting address is an absolute address. If the CEJ/MEJ switch is in the DISABLE position, this instruction performs as a 260 instruction. For information on the dual CP bit, refer to the 260x instruction.

262x Monitor Exchange Jump to MA



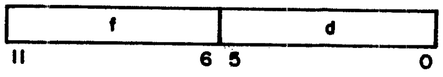
This 12-bit instruction is enabled or disabled by the CEJ/MEJ switch. If the switch is in the ENABLE position, this instruction is a conditional exchange jump of the CP. If the monitor flag bit clears, this instruction initiates the exchange jump and sets the flag. If the monitor flag sets, this instruction acts as a pass instruction. The starting address for this exchange jump is the 18-bit address held in the MA register of the CP. This starting address is an absolute address. If the CEJ/MEJ switch is in the DISABLE position, this instruction performs as a 260 instruction. For information on the dual CP bit, refer to the 260x instruction.

27x Read Program Address



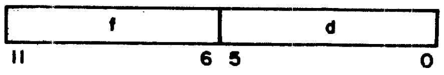
This 12-bit instruction transfers the content of the CP P register to the PP A register; this allows the PP to determine whether the CP is running. For information on the dual CP bit, refer to the 260x instruction.

30d Load (d)



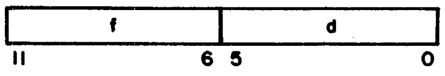
This 12-bit instruction clears the A register and loads the content of location d. The upper 6 bits of A are zero.

31d Add (d)



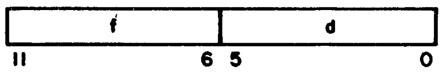
This 12-bit instruction adds to the A register the content of location d (treated as a 12-bit positive quantity).

32d Subtract (d)



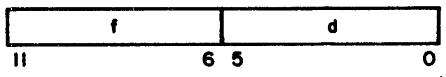
This 12-bit instruction subtracts from the A register the content of location d (treated as a 12-bit positive quantity).

33d Logical Difference (d)



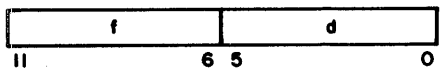
This 12-bit instruction forms in the A register the bit-by-bit logical difference of the lower 12 bits of A and the content of location d. This is equivalent to complementing individual bits of A which correspond to bits in location d that are ones. The upper 6 bits are not altered.

34d Store d



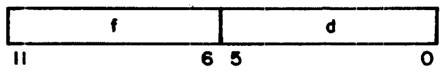
This 12-bit instruction stores the lower 12 bits of the A register in location d.

35d Replace Add (d)



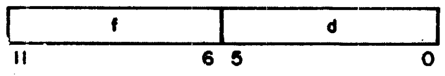
This 12-bit instruction adds the quantity in location d to the content of the A register and stores the lower 12 bits of the result in location d. The result is left in A at the end of the operation, and the original content of A is destroyed.

36d Replace Add One (d)



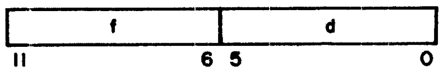
This 12-bit instruction replaces the quantity in location d with its original value plus 1. The result is left in the A register at the end of the operation, and the original content of A is destroyed.

37d Replace Subtract One (d)



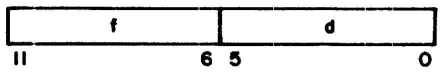
This 12-bit instruction replaces the quantity in location d with its original value minus 1. The result is left in the A register at the end of the operation, and the original content of A is destroyed.

40d Load ((d))



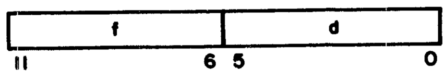
This 12-bit instruction clears the A register and loads a 12-bit quantity that is obtained by indirect addressing. The upper 6 bits of A are zero. Location d is read from PPM, and the word read is used as the operand address.

41d Add ((d))



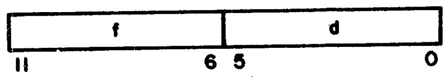
This 12-bit instruction adds to the content of the A register a 12-bit operand (treated as a positive quantity) obtained by indirect addressing. Location d is read from PPM, and the word read is used as the operand address.

42d Subtract ((d))



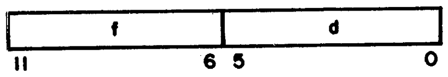
This 12-bit instruction subtracts from the A register a 12-bit operand (treated as a positive quantity) obtained by indirect addressing. Location d is read from PPM, and the word read is used as the operand address.

43d Logical Difference ((d))



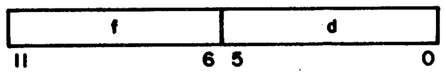
This 12-bit instruction forms in the A register the bit-by-bit logical difference of the lower 12 bits of A and the 12-bit operand read by indirect addressing. Location d is read from PPM, and the word obtained is used as the operand address. The upper 6 bits of A are not altered.

44d Store ((d))



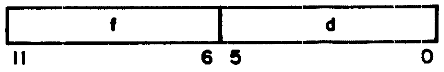
This 12-bit instruction stores the lower 12 bits of the A register in the location specified by the content of location d.

45d Replace Add ((d))



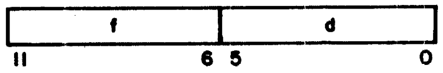
This is a 12-bit instruction. The operand, which is obtained from the location specified by the content of location d, is added to the content of the A register, and the lower 12 bits of the sum replace the original operand. The result is also left in A at the end of the operation.

46d Replace Add ((d))



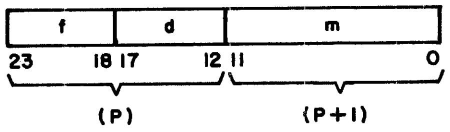
This is a 12-bit instruction. The operand, which is obtained from the location specified by the content of location d, is replaced by its original value plus 1. The result is also left in the A register at the end of the operation, and the original content of A is destroyed.

47d Replace Subtract One ((d))



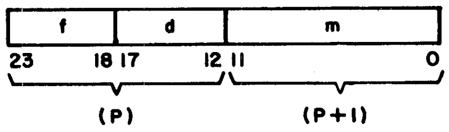
This is a 12-bit instruction. The operand, which is obtained from the location specified by the content of location d, is replaced by its original value minus 1. The result is also left in the A register at the end of the operation, and the original content of A is destroyed.

50dm Load (m + (d))



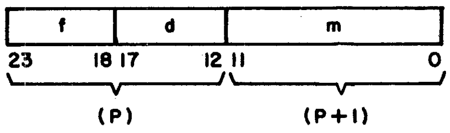
This 24-bit instruction clears the A register and loads a 12-bit quantity. The upper 6 bits of A are zeros. The 12-bit operand is obtained by indexed direct addressing. The quantity m, read from PPM location P plus 1, serves as the base operand address to which the content of d is added. If d equals 0, the operand address is m, but if d is not equal to 0, m plus the content in d is the operand address. Thus, location d may be used as an index quantity to modify operand addresses.

51dm Add (m + (d))



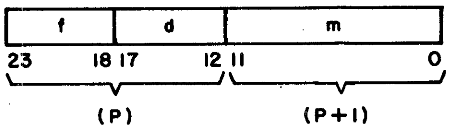
This 24-bit instruction adds to the A register the 12-bit operand (treated as a positve quantity) read by indexed direct addressing (refer to instruction 50).

52dm Subtract (m + (d))



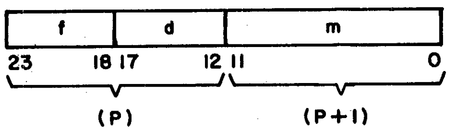
This 24-bit instruction subtracts from the A register the 12-bit operand (treated as a positive quantity) read by indexed direct addressing (refer to instruction 50).

53dm Logical Difference (m + (d))



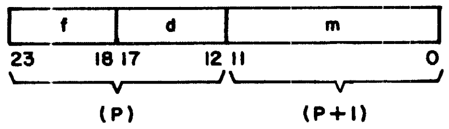
This 24-bit instruction forms in the A register the bit-by-bit logical difference of the lower 12 bits of A and a 12-bit operand obtained by indexed direct addressing. The upper 6 bits of A are not altered.

54dm Store (m + (d))



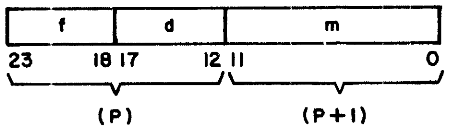
This 24-bit instruction stores the lower 12 bits of the A register in the location determined by indexed addressing (refer to instruction 50).

55dm Replace Add (m + (d))



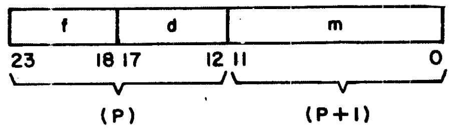
This is a 24-bit instruction. The operand, which is obtained from the location determined by indexed direct addressing, is added to the A register, and the lower 12 bits of the sum replace the original operand in PPM. The result is also left in A at the end of the operation, and the original content of A is destroyed.

56dm Replace Add One (m + (d))



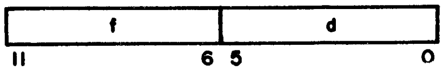
This is a 24-bit instruction. The operand, which is obtained from the location determined by indexed direct addressing, is replaced by its original value plus 1 (refer to instruction 50). The result is also left in the A register at the end of the operation, and the original content of A is destroyed.

57dm Replace Subtract One (m + (d))



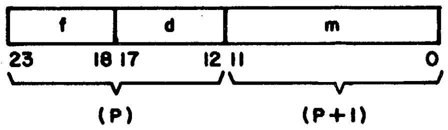
This is a 24-bit instruction. The operand, which is obtained from the location determined by indexed direct addressing, is replaced by its original value minus 1 (refer to instruction 50). The result is also left in the A register at the end of the operation, and the original content of A is destroyed.

60d Central Read from (A) to d



This 12-bit instruction transfers a 60-bit word from CM to five consecutive locations in the PPM. The 18-bit address of the CM location must be loaded into the A register prior to executing this instruction. (This is an absolute address.) The 60-bit word is disassembled into five 12-bit words beginning with the highest-order 12 bits. Location d receives the first 12-bit word. The remaining 12-bit words go to succeeding locations (d plus 1, d plus 2, and so on).

61dm Central Read (d) Words from (A) to m

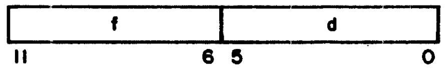


This 24-bit instruction reads a block of 60-bit words from CM. Location d contains the block length. The 18-bit address of the first central word must be loaded into the A register prior to executing this instruction. (This is an absolute address.) During the execution of the instruction, the content of P (P plus 1) goes to PP address 0 and m enters the P register. The content of d enters the Q register where it is reduced by one as each central word is processed. The content of address 0 is incremented by one and entered in the P register at the end of the instruction.

Each central word is disassembled into five 12-bit words beginning with the highest-order 12 bits. The first word is stored at PPM location m. The content of P (which is holding m) is advanced by one to provide the next address in the PPM as each 12-bit word is stored. If P overflows, operation continues as P is advanced from 7777_8 to 0000_8. These locations are written into as if they were consecutive. The data entered into location 0000 is one less than the address at which the PP resumes execution.

The content of A is advanced by one to provide the next CM address after each 60-bit word is disassembled and stored. Also, the content of the Q register is reduced by one. The block transfer is complete when Q equals zero. The block of CM locations goes from the address in A to the address in A plus the value in d minus 1. The block of PPM locations goes from address m to m plus 5 times the value in d minus 1.

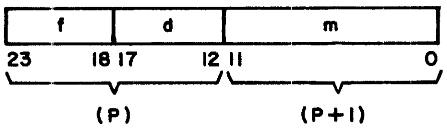
62d Central Write to (A) from d



This 12-bit instruction assembles five successive 12-bit words into a 60-bit word and stores the word in CM. The 18-bit address word designating the CM location must be in the A register prior to execution of the instruction. (This is an absolute address.)

Location d holds the first word to be read from the PPM. This word appears as the highest-order 12 bits of the 60-bit word to be stored in CM. The remaining words are taken from successive addresses.

63dm Central Write (d) Words to (A) from m



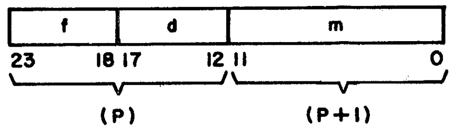
This 24-bit instruction assembles a block of 60-bit words and writes them in CM. Location d holds the number of 60-bit words. The A register holds the beginning CM address. (This is an absolute address.) During the execution of this instruction, the content of P (P plus 1) goes to PP address 0, and m enters the P register. The content of d enters the Q register, where it is reduced by one as each central word is assembled. The content of address 0 is incremented by one and entered in the P register at the end of the instruction.

The P register (the m portion of the instruction) holds the address of the first word to be read from the PPM. This word appears as the highest-order 12 bits of the first 60-bit word to be stored in CM.

P is advanced by one to provide the next address in the PPM as each 12-bit word is read. If P overflows, operation continues as P is advanced from 7777_8 to 0000_8. These locations are read as if they were consecutive. The data entered into location 0000 is one less than the address at which the PP resumes execution.

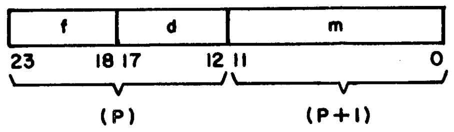
A is advanced by one to provide the next CM address after each 60-bit word is assembled. Also, Q is reduced by one. The block transfer is complete when Q equals zero.

64dm Jump to m if Channel d Active



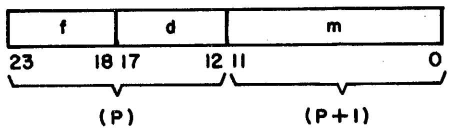
This 24-bit instruction provides a conditional jump to a new address specified by m. The jump is taken if the channel specified by d is active. The next instruction is at P plus 1, if the channel is inactive.

65dm Jump to m if Channel d Inactive



This 24-bit instruction provides a conditional jump to a new address specified by m. The jump is taken if the channel specified by d is inactive. The next instruction is at P plus 2 if the channel is active.

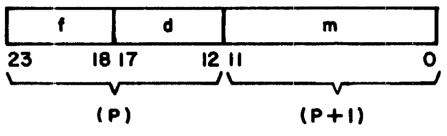
66dm Jump to m if Channel d Full



This 24-bit instruction provides a conditional jump to a new address specified by m. The jump is taken if the channel designated by d is full. The next instruction is at P plus 2 if the channel is empty.

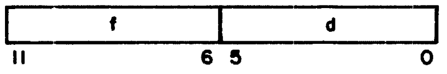
An input channel is full when the input equipment has placed a word in the channel and that word has not been accepted by a PP. The channel is empty when a word has been accepted. An output channel is full when a PP places a word on the channel. The channel is empty when the output equipment has accepted the word.

67dm Jump to m if Channel d Empty



This 24-bit instruction provides a conditional jump to a new address specified by m. The jump is taken if the channel specified by d is empty. The next instruction is at P plus 2 if the channel is full. (Refer to instruction 66 for explanation of full and empty.)

70d Input to A from Channel d

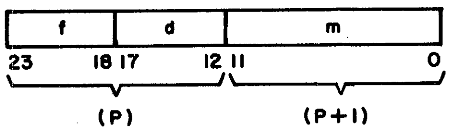


This 12-bit instruction transfers a word from input channel d to the lower 12 bits of the A register. The upper 6 bits of A are cleared to zero.

NOTE

If bit 5 of d is clear and the channel is inactive, this instruction hangs the PP, waiting for the channel to go active and full, if executed. If bit 5 of d is set and the channel is inactive or is deactivated before a full is received, the instruction exits. The word is not accepted, and the A register clears.

71dm Input (A) Words to m from Channel d



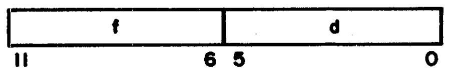
This 24-bit instruction transfers a block of 12-bit words from input channel d to the PPM. The first word goes to the PPM address specified by m. The A register holds the block length. The content of A is reduced by one as each word is read. The input operation is complete when A equals zero or the data channel becomes inactive. If the operation is terminated by the channel becoming inactive, the next storage location in the PPM is set to zero. However, the word count is not affected by this empty word. Therefore, the A register holds the block length minus the number of real data words read.

During this instruction, address 0000 temporarily holds P while m is held in the P register. P advances by one to hold the address for the next word as each word is stored.

NOTE

If this instruction is executed when the data channel is inactive, no input operation is accomplished, and the program continues at P plus 2. However, the location specified by m is set to zero. This exception is included to be compatible with existing CDC CYBER systems.

72d Output from A on Channel d

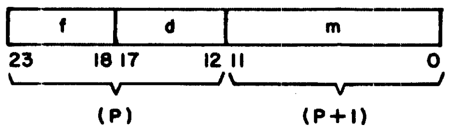


This 12-bit instruction transfers a word from the A register (lower 12 bits) to output channel d.

NOTE

If bit 5 of d is clear and the channel is inactive, this instruction hangs the PP, waiting for the channel to go active and full, if executed. If bit 5 of d is set and the channel is inactive, the program continues at P plus 1. The word is not transferred.

73dm Output (A) Words from m on Channel d



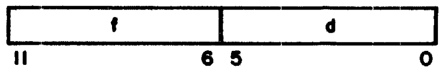
This 24-bit instruction transfers a block of words from the PPM to channel d. The first word is read from the address specified by m. The A register holds the number of words to be sent. A is reduced by one as each word is read. The output operation is complete when A equals zero or the channel becomes inactive.

During this instruction, address 0000 temporarily holds P while m is held in the P register. P advances by one to give the address of the next word as each word is read from the PPM.

NOTE

If this instruction is executed when the data channel is inactive, no output operation is accomplished, and the program continues at P plus 2.

74d Activate Channel d

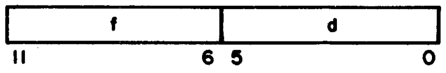


This 12-bit instruction activates the channel specified by d and sends the active signal on the channel to equipment connected to the channel. Activating a channel, which must precede a 70 through 73 instruction, prepares I/O equipment for the exchange of data.

NOTE

If this instruction is executed when the data channel is already active and if bit 5 of d is set, the program continues at P plus 1. Otherwise, activating an already active channel causes the PP to wait until the channel goes inactive. The PP hangs if the channel does not go inactive.

75d Disconnect Channel d



This 12-bit instruction deactivates the channel specified by d. As a result, the I/O data transfer stops.

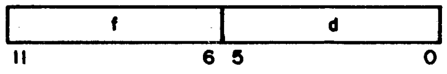
NOTES

1. If this instruction is executed when the data channel is already inactive and bit 5 of d is set, the program continues at P plus 1. Otherwise, deactivating an already inactive channel causes the PP to hang until the channel becomes active. The channel remains inactive, and no inactive signal is sent to the I/O equipment.

2. If an output instruction is followed by a disconnect instruction without first establishing that the information has been accepted by the input device (check for channel empty), the last word transmitted may be lost.

3. Do not deactivate a channel before putting a useful program in the associated PP. PPs other than 0 are hung on an input instruction (71) after deadstart. Deactivating a channel after deadstart causes an exit to the address specified by the content of location 0000 plus 1 and execution of that program. If the channel is deactivated without a valid program in that PP, the PP executes whatever program was left in PPM. Therefore, the PP could run wild.

76d Function (A) on Channel d

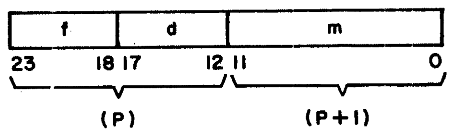


This is a 12-bit instruction. The external function code in the lower 12 bits of the A register is sent on channel d.

NOTE

If this instruction is executed with bit 5 of d clear and the channel active, PP execution stops until a deadstart or another PP causes the channel to become inactive. If bit 5 of d is set and the channel is active, the program continues at P plus 1. Neither the function signal nor the function word is transmitted. The channel remains active and execution continues.

77dm Function m on Channel d



This is a 24-bit instruction. The external function code specified by m is sent on channel d.

NOTE

If this instruction is executed with bit 5 of d clear and the channel active, PP execution stops until a deadstart or another PP causes the channel to become inactive. If bit 5 of d is set and the channel is active, the program continues at P plus 2. Neither the function signal nor the function word is transmitted. The channel remains active and execution continues.

PROGRAMMING INFORMATION 5

This section describes special programming information such as exchange jump, instruction execution, floating- and fixed-point arithmetic, address formats and data formats. The section also identifies status and control register bits and lists central processor (CP) error responses. Unless otherwise specified, all information in this section is applicable to all models.

CENTRAL PROCESSOR

EXCHANGE JUMP

An exchange jump instruction is an 013 instruction in the CP and a 26XX instruction in the peripheral processor subsystem (PPS). The instruction starts or interrupts the CP and provides central memory control (CMC) with the first address of a 16-word exchange package in central memory (CM). The address is K plus the content of Bj or the monitor address for a CP-initiated exchange. The address is the content of A of PPS-0 or PPS-1 or the content of MA in the PPS-initiated exchange. The PPS also has the monitor exchange jump to MA (262x) instruction in which the content of MA is used for the exchange address. The exchange package (Figure 5-1) provides the following information on a program to be executed.

Program address (P), 18 bits

Reference address for CM (RAC), 18 bits

Field length of program for CM (FLC), 18 bits

Exit mode (EM), 6 bits

Reference address for extended core storage (RAE), **21** bits

Field length of block transfer for extended core storage (FLE), 24 bits (lower 6 bits are assumed to be zeros)

Monitor address (MA), 18 bits

Initial contents of eight A registers, 18 bits

Initial contents of eight X registers, 60 bits

Initial contents of B1 through B7 (B0 constant 0) registers, 18 bits

The time during which a particular exchange package resides in the CP hardware registers is the execution interval. The execution interval begins with an exchange jump that swaps the exchange package information in CM with the information contained in the CP registers. The execution interval ends with the next exchange jump.

A hardware flag called a monitor flag (MF) indicates the type of program the CP is executing. When the flag is set, the program executing is a monitor program. When the flag is clear, the program executing is a user program.

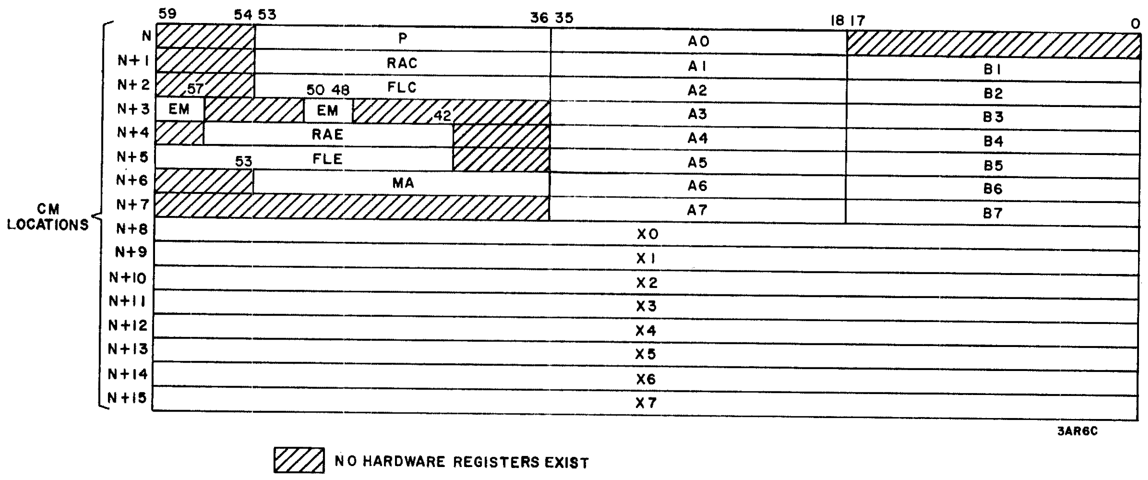


Figure 5-1. Exchange Package

A CP instruction and three peripheral processor (PP) instructions may initiate exchange jumps and select the exchange package that is to begin execution as follows:

CP instruction 013

PP instructions 2600, 2610, and 2620

The central exchange jump/monitor exchange jump (CEJ/MEJ) switch, located on the deadstart panel, enables or disables the CEJ/MEJ modes of operation. Following each change of the switch position, a deadstart is required before the change is recognized.

CEJ/MEJ Switch in DISABLE Position:

013 instruction	Handled as illegal instruction
2600, 2610, and 2620 instructions	Exchange jump to the address in A

CEJ/MEJ Switch in ENABLE Position:

013 instruction	If MF is clear, the starting address of the exchange package is the content of MA, and MF sets. If MF is set, the starting address for the exchange package is K plus the content of Bj, and MF clears.
2600 instruction	Exchange jump to the address in A.
2610 instruction	If MF is clear, the starting address of the exchange package is the content of A, and MF sets. If MF is set, the instruction acts as a pass instruction.
2620 instruction	If MF is clear, the starting address of the exchange package is the content of MA, and MF sets. If MF is set, the instruction acts as a pass instruction.

INSTRUCTION EXECUTION MODELS 172 THROUGH 174

Models 172 through 174 CPs sequentially read and execute program instruction words from their CMs. The CPs read the instructions by the use of a read next instruction (RNI) operation. The RNI operation is initiated between executions of the first and second instruction (Figure 5-2) of the current instruction word being processed and occurs partially during the time of the instruction executions. Initiating an RNI operation requires 2 clock periods. The remainder of the RNI operation takes place during the execution of the remaining instructions in the word being processed.

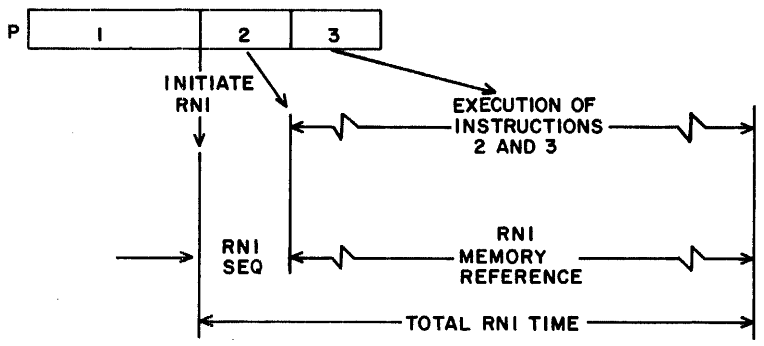


Figure 5-2. Instruction Execution Models 172 Through 174

In calculating execution times, 2 clock periods are added to each instruction word in a program to cover the RNI initiation time. The return jump and jump instructions are exceptions when their jump conditions are met and they occupy the upper positions of the instruction words. Since the times for these instructions already include the time required to read the new instruction word at the jump address, no additional time is consumed.

Example:

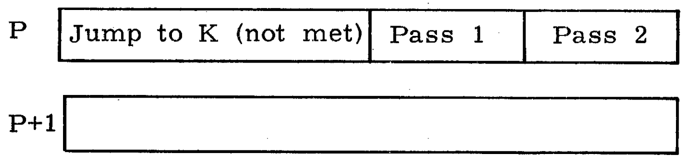
P	Jump to K (met)	Pass	Pass

K	Add 1	Add 2	Shift 1	Shift 2

Instruction	Model 173 Clock Periods Required
Jump	22
Add 1	6
RNI initiation	2
Add 2	6
Shift 1	7
Shift 2	7
Total time	50 clock periods

After RNI is initiated (between the first and second instructions of the word), a minimum of 18 clock periods elapses before the next instruction word is available for execution. Even if the lower-order positions of the word require less than 15 clock periods, a minimum of 18 clock periods is required.

Example:



For program optimization in the CP, always attempt to place jump and load/store instructions in the upper portion of the instruction word. The optimization avoids the additional time for RNI and the additional wait time required to allow the unwanted RNI (following a jump) to finish with CM. If a load/store instruction is placed in the second parcel and CM is not active, 4 clock periods are required for the RNI and return of the accept signal before initiation of the load/store instruction. If CM is active, more than 4 clock periods are required.

INSTRUCTION EXECUTION MODEL 175

Program instruction words are read one at a time from the instruction word stack (IWS) into the current instruction word (CIW) register for execution. An instruction issues from the CIW register when the conditions in the functional units and operating registers are such that the functions required for execution may be performed to completion without conflicting with a previously issued instruction. Once an instruction has issued, it must be completed in a fixed time frame. No delays are allowed from issue to delivery of data to the destination operating registers.

Since each instruction word is divided into four 15-bit parcels, as many as four instructions may be in the CIW register at one time. These instructions are executed in sequence (beginning with parcel 0). Allowance must be made for the mixture of one- and two-parcel instruction formats. Two-parcel instructions cannot be initiated in parcel 3.

When program execution reaches a branch instruction, the action taken depends upon whether the destination address is already in the instruction address stack (IAS). If the destination address is in the IAS, the P register is altered to the new program address and the corresponding word is read from the IWS to the CIW register. The jump is then completed without a CM reference for a new instruction word.

If the destination address is not in the IAS, two new words (located at the destination address and the destination address plus 1) are requested from CM to begin the new program sequence, and the stack is voided. Instruction execution continues upon receipt of the words from CM.

A branch out of the IWS may occur when the destination address corresponds to a program word that has already been requested from CM as a result of the sequential two-word read-ahead. If the word has not arrived at the IWS at the time of the branch test, the jump occurs, and the IWS is voided. If the word arrives before the branch test, the stack provides the word for execution, and the stack is not voided.

Because the IWS provides a copy of CM data for execution, it is necessary to ensure that the stack is voided when attempting instruction modification. The IWS can be voided by executing a return jump (01) instruction, long jump (02) instruction, or any branch (03 through 07) instruction to an address not in the stack.

FLOATING-POINT ARITHMETIC

Format

Floating-point arithmetic expresses a number in the form kB^n, where:

k	Coefficient
B	Base number
n	Exponent or power to which the base number is raised

B is assumed to be 2 for binary-coded quantities. In the 60-bit floating-point format (Figure 5-3), the binary point is considered to be to the right of the coefficient. The lower 48 bits express the integer coefficient, which is the equivalent of 15 decimal digits. The sign of the coefficient is separated from the rest of the coefficient and appears in the highest-order bit of the packed word. Negative numbers are represented in ones-complement notation.

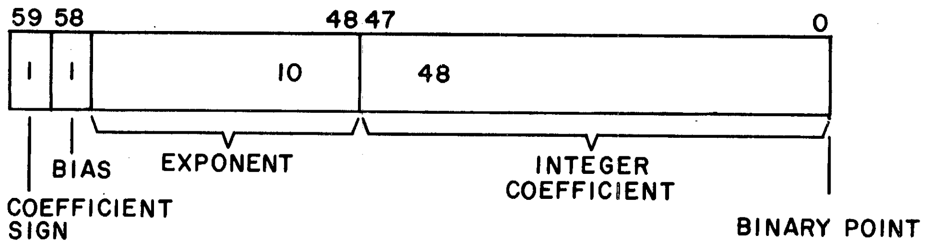


Figure 5-3. Floating-Point Format

The exponent is biased by complementing the exponent sign bit.

Table 5-1 summarizes the configurations of bits 58 and 59 and the implications, regarding signs, of the possible combinations.

TABLE 5-1. BITS 58 AND 59 CONFIGURATIONS

Bit 59	Bit 58	Coefficient Sign	Exponent Sign
0	1	Positive	Positive
0	0	Positive	Negative
1	0	Negative	Positive
1	1	Negative	Negative

Packing

Packing refers to the conversion of numbers in the form kB^n to floating-point format. A shortcut method of packing exponents can be derived by considering the representation of negative and positive zero exponents. Assuming a positive coefficient, zero exponents are packed as follows:

Positive zero exponent = 2000x----------x

Negative zero exponent = 1777x----------x

Since positive exponents are expressed in true form, begin with a bias of 2000 (positive zero) and add the magnitude of the exponent. The range of positive exponents is 0000 through 1777. In packed form, the range is 2000 through 3777.

When the coefficient is negative, the packed positive exponent is complemented to become 5777 through 4000.

Negative exponents are expressed in complement form by beginning with a bias of 1777 (negative zero) and then subtracting the magnitude of the exponent. The range of negative exponents is negative 0000 through negative 1777. In packed form, the range is 1777 through 0000.

When the coefficient is negative, the packed negative exponent is complemented to become 6000 through 7777.

Examples of packed and unpacked floating-point numbers are shown in octal notation to illustrate the packing process. Examples 1 and 2 are different forms of the integer positive 1. Example 3 is positive 100 (decimal), and example 4 is negative 100 (decimal). Examples 5 and 6 are large and small positive numbers. The unpacked values are shown as they might appear in the X and B registers prior to a pack operation.

Note that the packed negative zero exponent is not used for normal operation. Instead, 1777 is used to indicate the special error condition of indefinte.

```
1. Unpacked coefficient = 0000 0000 0000 0000 0001
   Unpacked exponent    =   00 0000
   Packed format        = 2000 0000 0000 0000 0001

2. Unpacked coefficient = 0000 4000 0000 0000 0000
   Unpacked exponent    =   77 7720
   Packed format        = 1720 4000 0000 0000 0000

3. Unpacked coefficient = 0000 6200 0000 0000 0000
   Unpacked exponent    =   77 7726
   Packed format        = 1726 6200 0000 0000 0000

4. Unpacked coefficient = 7777 1577 7777 7777 7777
   Unpacked exponent    =   77 7726
   Packed format        = 6051 1577 7777 7777 7777

5. Unpacked coefficient = 0000 4711 3000 0044 7021
   Unpacked exponent    =   00 1363
   Packed format        = 3363 4771 3000 0044 7021

6. Unpacked coefficient = 0000 6301 0277 4315 6033
   Unpacked exponent    =   77 6210
   Packed format        = 0210 6301 0277 4315 6033
```

Overflow

Overflow of the floating-point range is indicated by an exponent value of positive 1777 (3777 or 4000 in packed form). This is the largest exponent value that can be represented in the floating-point format. This exponent value may result from the calculation in which this exponent value, together with the computed coefficient value, is a correct representation of the result. This situation is called a partial overflow. However, further computation using this result generates an overflow.

A complete overflow occurs whenever a result requires an exponent larger than positive 1777. In this case, a complete overflow value results. This result has a positive 1777 exponent and a zero coefficient. The sign of the coefficient is the same as that which would have been generated if the result had not overflowed the floating-point range.

Underflow

Underflow of the floating-point range is indicated by an exponent value of negative 1777 (0000 or 7777 in packed form). This is the smallest exponent value that can be represented in the floating-point format. This exponent value may result from the calculation in which this exponent value, together with the computed coefficient value, is a correct representation of the result. This situation is called a partial underflow. Further computation using this result may be detected as an underflow.

A complete underflow occurs whenever a result requires an exponent smaller than negative 1777. In this case, a complete underflow value results. This result has a negative 1777 exponent and a zero coefficient. The complete underflow indicator is a word of all zeros. It is the same as a zero word in integer format.

Indefinite

An indefinite result indicator is generated whenever the calculation cannot be resolved. An example is division when the divisor is 0 and the dividend is also 0. Another example is multiplication of an overflow number times an underflow number. The indefinite result indicator is a value that cannot occur in normal floating-point calculations. This indicator corresponds to a negative 0 exponent and a 0 coefficient (177770 ------ 0 in packed form).

Any indefinite indicator used as an operand generates an indefinite result no matter what the other operand value is. Although indefinite indicators are always generated with a positive sign, they may occur as operands with a negative sign.

Nonstandard Operands

In summary, the special operand forms in octal are:

Positive overflow (+∞)	=	3777x------x
Negative overflow (-∞)	=	4000x------x
Positive indefinite (+IND)	=	1777x------x
Negative indefinite (-IND)	=	6000x------x
Positive underflow (+0)	=	0000x------x
Negative underflow (-0)	=	7777x------x

When one of these six special forms is used as an operand, only the following octal words can occur as results.

Positive overflow (+∞)	=	37770------0	Overflow condition flag
Negative overflow (-∞)	=	40000------0	Overflow condition flag
Indefinite (IND)	=	17770------0	Indefinite condition flag
Underflow (0)	=	00000------0	Underflow condition flag

Tables 5-2 through 5-5 indicate the resulting forms when various combinations of underflow, overflow, and indefinite forms are used in floating-point operations. The designations W and N are defined as follows:

W	Any word except ± ∞ and ± IND
N	Any word except ± ∞, ± IND, and ± 0

TABLE 5-2. Xj PLUS Xk (30, 32, 34 INSTRUCTIONS)

		Xk			
		W	+ ∞	- ∞	±IND
Xj	W		+ ∞	- ∞	IND
	+ ∞	+ ∞	+ ∞	IND	IND
	- ∞	- ∞	IND	- ∞	IND
	±IND	IND	IND	IND	IND

TABLE 5-3. Xj MINUS Xk (31, 33, 35 INSTRUCTIONS)

		Xk			
		W	+ ∞	- ∞	±IND
Xj	W		- ∞	+ ∞	IND
	+ ∞	+ ∞	IND	+ ∞	IND
	- ∞	- ∞	- ∞	IND	IND
	±IND	IND	IND	IND	IND

TABLE 5-4. Xj MULTIPLIED BY Xk (40, 41, 42 INSTRUCTIONS)

		Xk						
		+N	-N	+0	-0	+ ∞	- ∞	±IND
Xj	+N			0	0	+ ∞	- ∞	IND
	-N			0	0	- ∞	+ ∞	IND
	+0	0	0	Integer† Multiply		IND	IND	IND
	-0	0	0			IND	IND	IND
	+ ∞	+ ∞	- ∞	IND	IND	+ ∞	- ∞	IND
	- ∞	- ∞	+ ∞	IND	IND	- ∞	+ ∞	IND
	± IND	IND	IND	IND	IND	IND	IND	IND

†If both operands used in the integer multiply are normalized, an underflow results.

TABLE 5-5. Xj DIVIDED BY Xk (44, 45 INSTRUCTIONS)

		Xk						
		+N	-N	+0	-0	$+\infty$	$-\infty$	±IND
XJ	+N			$+\infty$	$-\infty$	0	0	IND
	-N			$-\infty$	$+\infty$	0	0	IND
	0	0	0	IND	IND	0	0	IND
	0	0	0	IND	IND	0	0	IND
	$+\infty$	$+\infty$	$-\infty$	$+\infty$	$-\infty$	IND	IND	IND
	$-\infty$	$-\infty$	$+\infty$	$-\infty$	$+\infty$	IND	IND	IND
	±IND	IND	IND	IND	IND	IND	IND	IND

Normalized Numbers

A normalized floating-point number has as large a coefficient and as small an exponent as possible. A floating-point number in packed format is normalized if the coefficient sign bit is different from bit 47. This condition indicates that the coefficient has been shifted to the left until bit 47 contains the most significant bit in the coefficient; therefore, the floating-point number has no leading sign bits in the coefficient. The normalized instructions perform the coefficient shift. The floating-multiply and floating-divide instructions deliver normalized results when provided with normalized operands. The floating-add instructions may deliver unnormalized results even when both operands are normalized. Therefore, it is necessary to perform the normalize operation after each sequence of floating-add or floating-subtract operations if the result is to be kept in a normalized form.

Rounding

Floating-point instructions are provided to round the results in single-precision computation. These instructions are executed in the same amount of time as the unrounded versions. The operands are modified to accomplish the rounding function. The amount of bias introduced by the rounding operation varies and is affected by the coefficient value in the operands. The descriptions of the round instructions define the effects of rounding in detail.

Double-Precision Results

The floating-point arithmetic instructions generate double-precision results. Use of unrounded instructions allows separate recovery of upper and lower half results with proper exponents. Rounded instructions allow only upper half results to be obtained. Two instructions, one single-precision and one double-precision, are required to retrieve an entire double-precision result.

To add or subtract two floating-point numbers, the coefficient having the smaller exponent enters the upper half of an accumulator and is right shifted by the difference of the exponents. Then the other coefficient is added into the upper half of the accumulator. The result is a double-length register with the format shown in Figure 5-4.

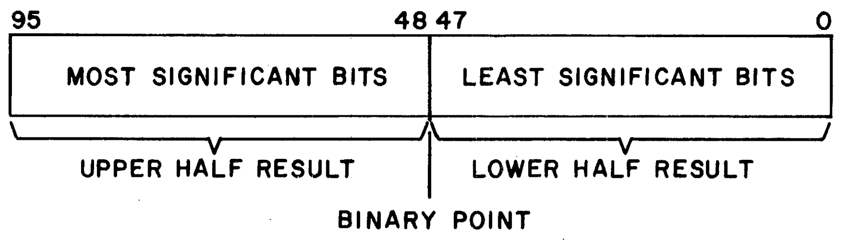


Figure 5-4. Floating-Add Result Format

If single precision is selected, the upper 48 bits of the 96-bit result and the larger exponent are returned as the result. Selecting double precision causes only the lower 48 bits of the 96-bit result and the larger exponent minus 60 (octal) to be returned as the result. The subtraction of 60 (octal) is necessary because the binary point is effectively moved from the right of bit 48 to the right of bit 0.

A 96-bit product is generated from two 48-bit coefficients. The result of a multiply is a double-length register with the format shown in Figure 5-5.

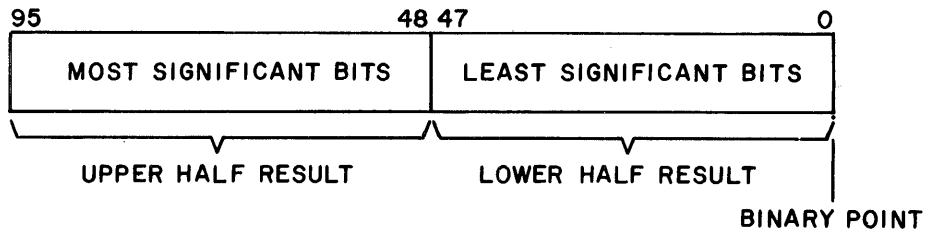


Figure 5-5. Multiply Result Format

If single precision is selected, the upper 48 bits of the product and the sum of the exponents plus 60 (octal) are returned as the result. The addition of 60 (octal) is necessary because the binary point is effectively moved from the right of bit 0 to the right of bit 48 when the upper half of the 96-bit result is selected. If double precision is selected, the result is the lower 48 bits of the product and the sum of the exponents.

FIXED-POINT ARITHMETIC

Fixed-point addition and subtraction of 60-bit numbers are handled by the long-add instructions (36, 37). Negative numbers are represented in ones-complement notation, and overflows are ignored. The sign bit is in the high-order bit position (bit 59), and the binary point is to the right of the low-order bit position (bit 0).

Fixed-point addition and subtraction of 18-bit numbers are handled by the increment instructions (50 through 77). Negative numbers are represented in ones-complement notation, and overflows are ignored. The sign bit is in the high-order bit position (bit 17), and the binary point is to the right of the low-order bit position (bit 0).

Integer multiplication is handled as a subset operation of the floating-multiply (42) instruction. The integer multiply requires that both 47-bit integer operands have zero exponents and are not normalized. The result is 48 bits with sign extension. Normalized operands cause underflow results to be reported. If the results exceed 48 bits, overflow is not detected.

An integer divide takes several steps. For example, an integer quotient X1 equal to X2/X3 is produced by the following steps.

	Instructions	Remarks
1.	Pack X2 from X2 and B0	Pack X2
2.	Pack X3 from X3 and B0	Pack X3
3.	Normalize X3 in X0 and B0	Normalize X3 (divisor)
4.	Floating quotient of X2 and X0 to Xi	Divide
5.	Unpack X1 to X1 and B7	Unpack quotient
6.	Shift X1 nominally left B7 places	Shift to integer position

The divide requires that:

1. Both integer (2^{47} maximum) operands be in floating-point format, and
2. The divisor be left-shifted 48 places, or
3. The quotient be right-shifted 48 places, or
4. Any combination of n left-shifts of the divisor and 48 minus n right-shifts of the quotient be accomplished.

The normalize X3 instruction left shifts the divisor n places ($n \geq 0$), providing a divisor exponent of negative n. The quotient exponent is then $0 - (-n) - 48 = n - 48 \leq 0$.

After unpacking and left shifting nominally, the negative (or zero) value in B7 right-shifts the quotient 48 minus n places, producing an integer quotient in X1. A remainder may be obtained by an integer multiply of X1 and X3 and subtracting the result from X2.

COMPARE/MOVE ARITHMETIC MODELS 172 THROUGH 174

The compare/move arithmetic provides multiple character manipulation. The characters are six bits in length. Characters can be moved from one CM location to another, and fields of characters can be compared either directly or through a collation table.

The move direct instruction moves a field of up to 127 characters from one location to another location as specified in the instruction. The move indirect instruction performs the same kind of move, but a CM reference is used to obtain the parameters. The move indirect instruction moves a field of up to 8181 characters.

The compare collated instruction compares two fields of up to 127 characters. When two characters are found to be unequal, the characters are referenced in a collation table and the values found are compared. If those values are unequal, the field with the larger character is indicated. The compare uncollated instruction compares two fields of up to 127 characters and indicates the larger of the first character pair which is found to be unequal.

CENTRAL PROCESSOR DIFFERENCES

Differences exist betwen the models 172 through 174 and the model 175 floating-multiply and floating-add/subtract operations.

Multiply Difference

A difference exists when an exponent overflow of a floating product occurs and the coefficient result requires a left shift of one to give a normalized answer. The model 175 tests for the overflow condition by checking for the exponent being greater than positive 1777 before correction, if any, is made for a left shift of one. Thus, even though the left shift of one may cause the exponent to equal positive 1777 (partial overflow), this condition is treated as a complete overflow, and the result is the overflow exponent with a zero coefficient. Models 172 through 174 test for the overflow condition by checking for the exponent greater than positive 1777 after correction, if any, is made for a left shift of one. In this case, if the resulting exponent is positive 1777 (partial overflow), the result is the overflow exponent with the computed coefficient.

```
Example:  40012

    X1 = 3700 4000 0000 0000 0000
    X2 = 2020 4000 0000 0000 0000

Model 175 result:                X0 = 3777 0000 0000 0000 0000
Models 172 through 174 result:   X0 = 3777 4000 0000 0000 0000
```

A similar situation exists when an exponent underflow of a floating product occurs and the coefficient result does not require a left shift of one to give a normalized answer. The model 175 tests for the underflow condition by checking for the exponent being less than negative 1777 before correction, if any, is made for a left shift of one. Although no left shift of one is performed, an exponent of negative 1777 (partial underflow) is treated as a complete underflow, and the result is the underflow condition with zero coefficient. Models 172 through 174 test for the underflow condition by checking for the exponent less than negative 1777 after correction, if any, is made for a left shift of one. In this case, if the resulting exponent is negative 1777 (partial underflow), the result is the underflow exponent with the computed coefficient.

Example: 40012

```
    X1 = 0647 7777 7777 7777 7776
    X2 = 1050 4444 4444 4444 4444

Model 175 result:              X0 = 0000 0000 0000 0000 0000
Models 172 through 174 result: X0 = 0000 4444 4444 4444 4442
```

Floating-Add Differences

A difference exists when an exponent underflow of a floating double-precision sum occurs and the coefficient result requires a right shift of one because coefficient overflow occurred. The model 175 tests for the underflow condition by checking for the exponent being less than negative 1777 before correction, if any, is made for a right shift of one. Thus, even through the right shift of one may cause the exponent to equal negative 1777 (partial underflow), this condition is treated as a complete underflow, and the result is the underflow exponent with a zero coefficient. Models 172 through 174 test for the exponent underflow condition by checking for the exponent less than negative 1777 after correction, if any, is made for a right shift of one. In this case, if the resulting exponent is negative 1777 (partial underflow), the result is the underflow exponent with the computed coefficient.

Example: 32012

```
    X1 = 0057 4000 0000 0000 0001
    X2 = 0057 4000 0000 0000 0000

Model 175 result:              X0 = 0000 0000 0000 0000 0000
Models 172 through 174 result: X0 = 0000 4000 0000 0000 0000
```

ILLEGAL INSTRUCTIONS

The following instructions cause an error exit to MA or program stop. System error responses for illegal instructions are listed in Tables 5-7, 5-8, and 5-9. In addition to causing error responses, illegal instructions are executed as passes and do not change the content of any register (except as noted in 5 of the following list).

1. 011, 012 with no ECS or in parcel 1, 2, 3
2. 013 with CEJ/MEJ disabled or in parcel 1, 2, 3

3. 014 through 017
4. 464 through 467 (model 175), 464 through 467 in parcel 1, 2, 3 (models 172 through 174)
5. Any 30-bit instruction in parcel 3. (In models 172 through 174, these illegal instructions execute. The lower 15 bits of the instruction are provided by whatever bits are in that part of the instruction register. Once into execution, the instruction is detected as illegal and aborted. Registers and P values are changed before the program is stopped.)

CENTRAL PROCESSOR EXIT MODE/ERROR RESPONSE

When the CP detects or is informed of an error, it records the error. Depending upon the type of error and the mode selection bits, the program in execution may be interrupted. If the error is an illegal instruction, breakpoint, or an address-range error on an RNI or branch, the program interruption is unconditional. For other types of errors, the mode selection bits determine whether or not the program is interrupted. If the mode selection bit is set and the corresponding condition is detected, the program is interrupted. These sections are contained in word N plus 3 of the exchange package and are selected as shown in Table 5-6.

TABLE 5-6. CP PROGRAM INTERRUPT CONDITIONS

Condition Bit	Mode Selection Bit	Interrupt Condition
48	48	Address range error
49	49	Infinite mode
50	50	Indefinite mode
51	57	Parity error on ECS flag register operation
52	58	CMC input error
53	59	CM data error

Error conditions 48, 49, and 50 are detected in the CP, and conditions 51, 52, and 53 are flags sent to the CP from the CMC. The parity error on ECS flag register operations indicates a transmission error on the address between the ECS coupler and ECS controller when the ECS flag register operation is being used. The CMC input error flag indicates that a transfer from the CP caused a data or address parity error at CMC or an address parity error at CM. The CM data error indicates a double error on data requested by the CPU in SECDED mode of operation or a CM data parity error in a parity mode of operation.

Any error condition detected after an exchange jump instruction has started execution is treated as an error for the incoming program. Figure 5-6 shows the format of relative address zero on an error exit.

Tables 5-7 through 5-9 explain what happens when the various kinds of errors occur. The tables list the same error conditions with different CEJ/MEJ or MF conditions. The error response depends upon the setting of the CEJ/MEJ switch and the state of the MF. The table headings specify the three combinations.

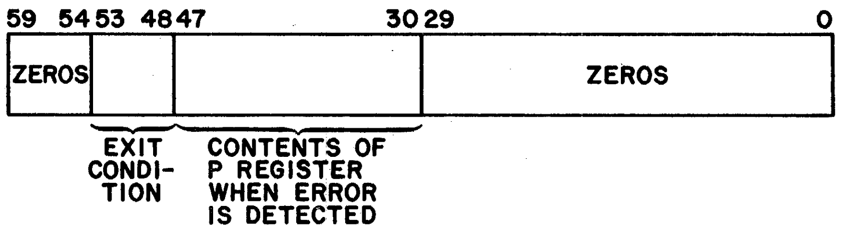


Figure 5-6. Format of Relative Address Zero on Error Exit

TABLE 5-7. ERROR RESPONSE WITH CEJ/MEJ ENABLED, MF SET

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Illegal instruction	1. Execute the illegal instruction as if it were a pass. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Execute the illegal instruction as if it were a pass. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.
Exit condition bit 48 set by an increment read of an address out of range	1. Read all zeros to the selected X register. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Read all zeros to the selected X register. 2. Continue execution
Exit condition bit 48 set by an increment write of an address out of range	1. Block write operation, content of CM is unchanged. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Block write operation, content of CM is unchanged. 2. Continue execution.
Exit condition bit 48 set on RNI or branch out of range	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.
Exit condition bit 48 set on CMU instruction (models 172 through 174 only) 1. C1 or C2 > 9 2. K1 or K2 address out of range	1. Error condition 1 causes instruction to execute as pass. Condition 2 causes instruction moves or compares up to the point of address out of range. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Error condition 1 causes instruction to execute as pass. Condition 2 causes instruction moves or compares up to the point of address out of range. 2. Continue with next 60-bit instruction.
Exit condition bit 48 set by an ECS address range check	1. Force ECS instruction to execute as a pass instruction. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Force ECS instruction to execute as a pass instruction. 2. Exit to next 60-bit word. 3. Continue execution with next 60-bit word.
Infinite condition (bit 49) Indefinite condition (bit 50) ECS flag register parity (bit 51) CM data error condition (bit 53)	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.	Continue execution.
CMC input error condition (bit 52)	1. Block write operation, content of CM is unchanged. 2. Block read operation forces read data to all ones. 3. Stop CP. 4. Store P and exit condition bits at RAC. 5. Clear P.	1. Block write operation, content of CM is unchanged. 2. Block read operation forces read data to all ones. 3. Continue execution.

TABLE 5-7. ERROR RESPONSE WITH CEJ/MEJ ENABLED, MF SET (Cont'd)

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
00 instruction	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.
Breakpoint signal from CMC (refer to breakpoint notes)	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.

TABLE 5-8. ERROR RESPONSE WITH CEJ/MEJ ENABLED, MF CLEAR

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Illegal instruction	1. Execute the illegal instruction as if it were a pass. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Execute the illegal instruction as if it were a pass. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.
Exit condition bit 48 set by an increment read of an address out of range	1. Read all zeros to the selected X register. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Read all zeros to the selected X register. 2. Continue execution.
Exit condition bit 48 set by an increment write of an address out of range	1. Block write operation, content of CM is unchanged. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Block write operation, content of CM is unchanged. 2. Continue execution.
Exit condition bit 48 set by an RNI or branch address out of range	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.

TABLE 5-8. ERROR RESPONSE WITH CEJ/MEJ ENABLED, MF CLEAR (Cont'd)

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Exit condition bit 48 set on CMU instruction (models 172 through 147 only) 1. C1 or C2 > 9 2. K1 or K2 address out of range	1. Error condition 1 causes instruction to execute as pass. Condition 2 causes instruction moves or compares up to the point of address out of range. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Error condition 1 causes instruction to execute as pass. Condition 2 causes instruction moves or compares up to the point of address out of range. 2. Continue with next 60-bit instruction.
Exit condition bit 48 set by an ECS address range check	1. Force ECS instruction to execute as a pass instruction. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Force ECS instruction to execute as a pass instruction. 2. Continue execution with next 60-bit word.
Infinite condition (bit 49) Indefinite condition (bit 50) ECS flag register parity (bit 51) CM data error condition (bit 53)	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.	Continue execution.
CMC input error condition (bit 52)	1. Block write operation, content of CM is unchanged. 2. Block read operation forces read data to all ones. 3. Stop CP. 4. Store P and exit condition bits at RAC. 5. Clear P. 6. Exchange jump to MA and set MF.	1. Block write operation, content of CM is unchanged. 2. Block read operation forces read data to all ones. 3. Continue execution.
00 instruction	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P. 4. Exchange jump to MA and set MF.
Breakpoint signal from CMC (refer to breakpoint notes)	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.	1. Execute remaining parcels of 60-bit word currently executing. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P. 5. Exchange jump to MA and set MF.

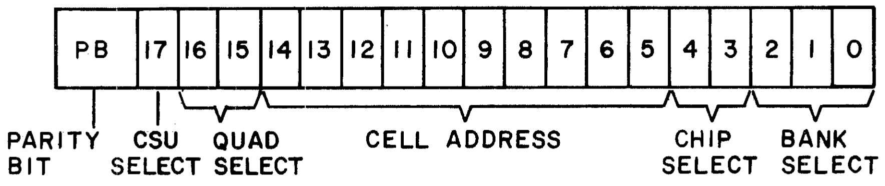
TABLE 5-9. ERROR RESPONSE WITH CEJ/MEJ DISABLED

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Illegal instruction	1. Execute the illegal instruction as if it were a pass. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Execute the illegal instruction as if it were a pass. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.
Exit condition bit 48 set by an increment read of an address out of range	1. Read all zeros to the selected X register. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Read all zeros to the selected X register. 2. Continue execution.
Exit condition bit 48 set by an increment write of an address out of range	1. Block write operation, content of CM is unchanged. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Block write operation, content of CM is unchanged. 2. Continue execution.
Exit condition bit 48 set by an RNI or branch address out of range	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.	Stop CP.
Exit condition bit 48 set on CMU instruction 1. C1 or C2 > 9 2. K1 or K2 address out of range	1. Error condition 1 causes instruction to execute as pass. Condition 2 causes instruction moves or compares up to the point of address out of range. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Error condition 1 causes instruction to execute as pass. Condition 2 causes instruction moves or compares up to the point of address out of range. 2. Continue with next 60-bit instruction.
Exit condition bit 48 set by ECS address range check	1. Force ECS instruction to execute as a pass. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Force ECS instruction to execute as a pass. 2. Continue execution with next 60-bit word.
Infinite condition (bit 49) Indefinite condition (bit 50) ECS flag register parity (bit 51) CM data error condition (bit 53)	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.	Continue execution.
CM input error condition (bit 52)	1. Block write operation, content of CM is unchanged. 2. Block read operation forces read data to all ones. 3. Stop CP. 4. Store P and exit condition bits at RAC. 5. Clear P.	1. Block write operation, content of CM is unchanged. 2. Block read operation forces read data to all ones. 3. Continue execution.
00 instruction	Stop CP.	Stop CP.
Breakpoint signal from CMC (refer to breakpoint notes)	1. Execute remaining parcels of 60-bit instruction word. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Execute remaining parcels of 60-bit instruction word. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.

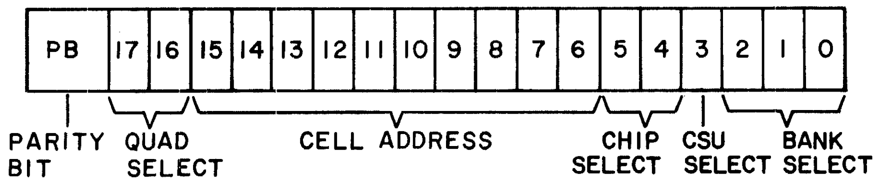
CENTRAL MEMORY

ADDRESS FORMAT

The 18-bit CM address is partially translated in CMC to a 14-bit address and 16 separate go bank signals. The translation is somewhat different for the models 172 through 174 and the model 175. In models 172 through 174, bits 0, 1, 2, and 17 are used for bank selection. In model 175, bits 0, 1, 2, and 3 are used for bank selection. In each case, the most significant bank select bit selects one of the two CSU chassis. The address formats are shown in Figure 5-7.



MODELS 172 THROUGH 174



MODEL 175

Figure 5-7. Central Memory Address Formats

DATA FORMAT

CM is capable of sending and receiving 68 bits of information. The 68 bits are comprised of 60 bits of data plus 8 SECDED code bits which are added and checked as the data is passed through CMC. The data format is shown in Figure 5-8.

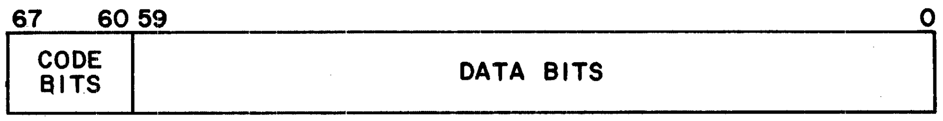


Figure 5-8. Data Format

BREAKPOINT

The breakpoint feature provides a diagnostic aid by allowing a breakpoint on a given absolute CM address.

An 18-bit field in the status and control register is reserved for the breakpoint address. Four additional bits specify control when breakpoint is enabled.

When a breakpoint compare occurs in CMC, the breakpoint flag is set and a signal is sent to the requesting unit. CM access is not blocked. The CMC reports the breakpoint status code to the status and control register.

Status and control register bit 77 is the CMC breakpoint match. This bit loads and locks bits 56 through 59 which hold the port code and condition code that resulted in breakpoint compare.

When a breakpoint compare occurs during a PPS access to CM, the breakpoint flag is sent to the PPS port. The PPS sets bit 76 of the status and control register to indicate that a PPS compare occurred. This bit locks in bits 60 through 75. If bit 83 is set, the PP number code is stored in bits 72 through 75 and the content of that PP's P register is stored in status and control register bits 60 through 71. This status is held until bit 76 clears.

The following breakpoint notes apply only to model 175.

1. Since breakpoint is for an address request to CM, a breakpoint does not occur for an instruction executed from the instruction stack if the instruction entered the instruction stack before selecting breakpoint.
2. The value of P plus RAC when the CP is stopped by breakpoint may not correspond with the value of breakpoint address because the CP normally requests two words ahead of P on an RNI.
3. The value of P plus RAC when the CP stops for a breakpoint on an increment address may not correspond with the value of P plus RAC of the increment instruction. Advancing P is based on the 60-bit word of instructions entering CIW instead of any given parcel of CIW being executed.

CENTRAL MEMORY PROTECTION

All references to CM from the CP for instructions or read/write data are made relative to RAC. RAC defines the lower limit of a CM program. Changes to RAC permit relocation of programs in CM.

During an exchange jump, an 18-bit RAC and an 18-bit FLC are loaded into their respective registers to define the CM limits of the program initiated by the exchange jump.

The relationship between absolute memory address, relative memory address, RAC, and FLC is indicated in Figure 5-9.

The following relationships must be true if the program is to operate within its bounds.

$\text{RAC} \leq (\text{RAC} + \text{P}) < (\text{RAC} + \text{FLC})$ (absolute memory addresses), or

$0 \leq \text{P} < \text{FLC}$ (relative memory addresses)

NOTE

To avoid possible artificial range faults, instructions should not be stored near the upper limit address of the field length. For example, using absolute address ((RAC + FLC) - 1) for an instruction produces a range fault when the RNI occurs to (RAC + FLC). Data, rather than instructions, should always be stored in addresses of absolute locations ((RAC + FLC) - 1) and (RAC + FLC).

CM references beyond the previously described limits cause error responses listed in Tables 5-7 through 5-9.

PERIPHERAL PROCESSOR SUBSYSTEM

The PPs have access to all CM storage locations. One 60-bit word or a block of 60-bit words can be transferred from a peripheral processor memory (PPM) to CM or from CM to PPM. (Five 12-bit PP words equal one 60-bit CM word.) Data from external devices is read into a PPM, and with additional instructions, is transferred to CM. Conversely, data is transferred from CM to a PPM and is then transferred, by additional instructions, to external devices. All addresses sent to CM from PPs are absolute addresses.

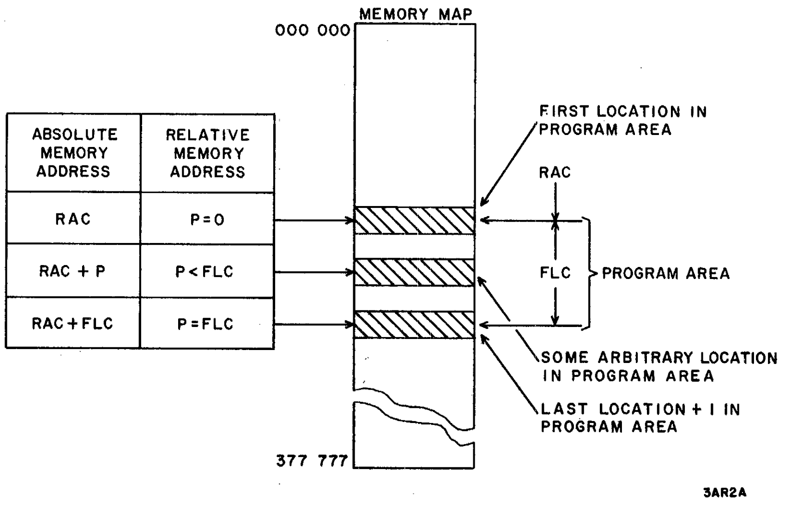


Figure 5-9. Memory Map

CENTRAL MEMORY READ

The CM words are delivered to a five-stage read pyramid where they are disassembled into five 12-bit words. A read pyramid exists in PPS-0 and in optional PPS-1, when installed.

At a 1X PP operating speed, one 12-bit word is transferred to a PP every microsecond. Because the CM word is 60 bits long, 5 microseconds are required for the transfer of each CM word. As many as four PPs can time-share the pyramid so that the transfer rate can be increased to four CM words each 5 microseconds.

If more than four PPs in one PPS are simultaneously requesting CM read operations, the instructions are maintained until the pyramid can accept another PP. Then the PPs are accepted in the order in which they appear. A waiting PP is not locked out of the pyramid.

The CM starting address must be entered in the A register before a read instruction can be executed. A load dm (20) instruction may be used.

For a one-word transfer, the d portion of the read (60) instruction specifies the following.

d is the PPM address (0000 through 0077_8) for the first 12-bit word. The remaining words go to locations d plus 1, d plus 2, and so on.

For a block transfer, the d and m portions of the read (61) instruction specify the following.

(d) is the number of CM words to be transferred.

m is the PPM first word address. The value in the A register is increased by one with the transfer of each word to locate consecutive CM words.

CENTRAL MEMORY WRITE

The 62 instruction is used for one word and the 63 instruction is used for a block transfer. These instructions assemble 12-bit words into 60-bit words and write them in CM. This assembly is performed in a write pyramid and then transferred to CM. A write pyramid exists in PPS-0 and in optional PPS-1, when installed. The read and the write pyramids can be time-shared by up to four PPs in one PPS. Write pyramid timing is similar to read pyramid timing.

The starting address in CM has to be entered in the A register before the write instruction is executed.

For a one-word transfer, the d portion of the write (62) instruction specifies the following.

d is the PPM address (0000 through 0077_8) of the first 12-bit word. The remaining words are taken from d plus 1, d plus 2, and so on.

For a block transfer, the d and m portions of the write (63) instruction specify the following.

(d) is the number of CM words to be transferred.

m is the PPM starting address. The value in the A register is increased by one with the transfer of each word to provide consecutive CM locations.

CHANNEL CONFLICTS IN AN EXPANDED PP SYSTEM

Channel conflicts can occur in a 14-, 17-, or 20-PP system when the PPs operate at a 2X speed and specific PPs attempt to access specific channels at the same time. The conflicts do not occur during a 1X operating speed. (Refer to Barrel and Slot in section 2.)

When a channel conflict exists, external and internal PP channels are slowed from the transfer rate of 2X operating speed to the transfer rate of 1X operating speed. The channels affected are as follows:

- External I/O channels (octally numbered 0 through 13 for PPS-0 and 20 through 33 for PPS-1)
- Internal status and control register channels (octally numbered 16 for PPS-0 and 36 for PPS-1)
- Internal real-time clock channel (octally numbered 14)
- Channels (octally numbered) 15 and 35 are not used and channels 17 and 37 do not exist

When an expanded PP system operates at a 1X speed, external channel requests (Figure 5-10) are timed to arrive at the opposite PPS chassis when the slot (refer to Barrel and Slot in section 2) is not referencing a channel. This timing ensures constant channel accesses.

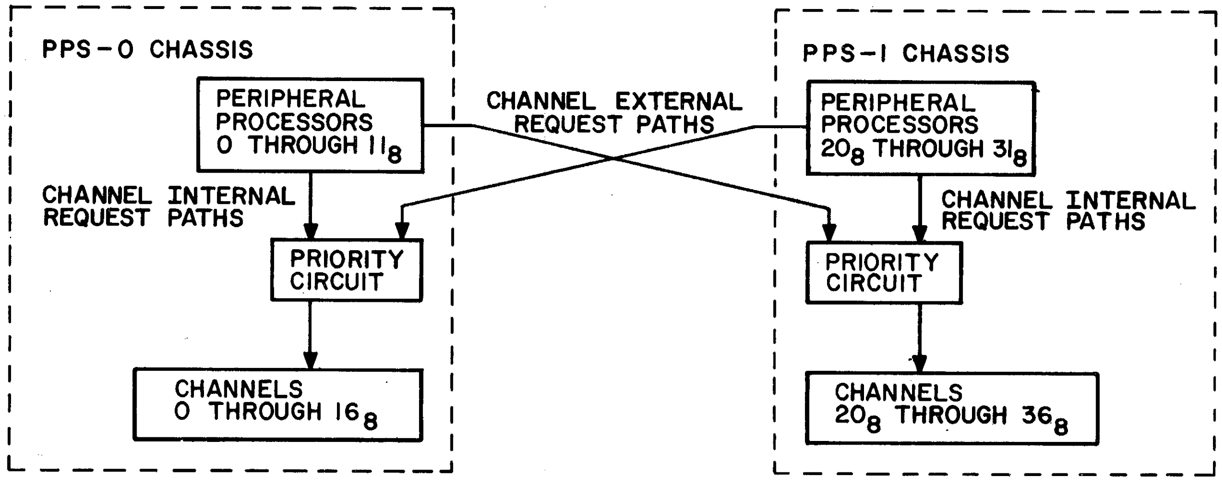

PPS-0 CHASSIS
PERIPHERAL PROCESSORS 0 THROUGH 11_8
CHANNEL INTERNAL REQUEST PATHS
PRIORITY CIRCUIT
CHANNELS 0 THROUGH 16_8
CHANNEL EXTERNAL REQUEST PATHS
PPS-1 CHASSIS
PERIPHERAL PROCESSORS 20_8 THROUGH 31_8
CHANNEL INTERNAL REQUEST PATHS
PRIORITY CIRCUIT
CHANNELS 20_8 THROUGH 36_8

Figure 5-10. PP Channel Request Paths

When the PPs operate at a 2X speed, the slot is continuously referencing PPs. Channel requests from external PPs are then allowed on the basis of internal requests or a 500-nanosecond toggle. The external requests are always honored if the internal PP which is being referenced by the slot at the time of the request is not performing channel instructions. A 2X transfer rate can then be maintained by the PP providing the external request, if a memory refresh does not cause a conflict. Only when the internal PP requests access the same group of channels as the external PP does the priority circuit slow the requesting internal and external PPs to the 1X transfer rate.

INPUT/OUTPUT TRANSFERS

Data Input Sequence

Input equipment sends data to the PP by way of the controller as follows:

1. The PP places a function word in the channel register and sets the full flag and the channel active flag. At the same time, the PP sends the first of a group of words and function signals to all controllers. The function signals cause all controllers to sample the words and identify the words as function codes rather than data words. Connect codes select controllers and modes of operation and clear nonselected controllers. Only selected controllers are connected.
2. The controller sends an inactive signal to the PP, indicating acceptance of the function code. The signal drops the channel active flag which in turn drops the full flag and clears the channel register.
3. The PP sets the channel active flag and sends an active signal to the controller which signals input equipment to start sending data.
4. The input equipment reads a 12-bit word plus 1 parity bit and then sends the word with parity to the channel register with a full signal which sets the channel full flag (10 to 15 nanoseconds after the data arrives).
5. The PP stores the word, drops the full flag, and returns an empty signal, indicating acceptance of the word. The input equipment clears its data register and prepares to send the next word.
6. Steps 4 and 5 are repeated for each word transferred.
7. At the end of the transfer, the controller clears its active condition and sends an inactive signal to the PP to indicate the end of data. The signal clears the channel active flag to disconnect the controller and the PP from the channel.

8. As an alternative, the PP may choose to disconnect from the channel before the input equipment has sent all of its data. The PP does this by dropping the active flag and sending an inactive signal to the controller which immediately clears its active condition and sends no more data, although the input equipment may continue to the end of its record or cycle. For example, a magnetic tape unit continues to the end of its record and stops in the record gap.

Data Output Sequence

The PP sends data to output equipment as follows:

1. The PP places a function word in the channel register and sets the full flag and the channel active flag. At the same time, the PP sends the first of a group of words and function signals to all controllers. The function signals cause all controllers to sample the words and identify the words as function codes rather than data words. Connect codes select controllers and modes of operation and clear nonselected controllers. Only selected controllers are connected.
2. The controller sends an inactive signal to the PP, indicating acceptance of the function code. The signal drops the channel active flag which in turn drops the full flag and clears the channel register.
3. The PP sets the channel active flag and sends an active signal to the controller which signals output equipment that data flow is starting.
4. The PP places a 12-bit data word plus 1 parity bit in the channel register and sets the full flag. At the same time, the PP sends the word with parity and a full signal to the controller.
5. The controller accepts the word and sends an empty signal to the PP where the signal clears the channel register and drops the full flag.
6. Steps 4 and 5 are repeated for each PP word.
7. After the last word is transferred and acknowledged by the controller empty signal, the PP drops the channel active flag and turns off the controller with an inactive signal.

FORCE PERIPHERAL PROCESSOR EXIT

If bit 124 is set, bit 125 in the status and control register causes the PP selected by bits 120 through 124 to do a forced instruction exit. The selected PP completes the current instruction and goes on to the next instruction without waiting for conditional replies. If bit 124 is clear, the PP is manually selected.

DEADSTART

Bit 126 in the status and control register causes the PP selected by bits 120 through 124 to be forced into a deadstart mode, waiting for input on its assigned channel (channel N for PP N). An active PP can then transmit a new program to the hung PP and restart the PP. This feature forces one PP to deadstart without disturbing the others and is used to unhang a PP.

RELEASE HUNG PERIPHERAL PROCESSOR TO CENTRAL MEMORY REQUEST

Bit 78 in the status and control register permits the unanswered CM reference to be bypassed by clearing central busy. This feature allows recovery from a lost accept signal on a PP to CM request. Bit 79 in the status and control register acts as a C5 full signal (received data from CM) and unhangs a PP that was hung on a CM read.

STATUS AND CONTROL REGISTER

The status and control register is a program-controlled register that monitors system error conditions and provides control of some system features. Bit assignments within the register permit monitoring of parity error and SECDED networks and for controlling such things as breakpoint, PP speed (1X or 2X), and maintainability features. In addition, the register provides control for testing the parity error and SECDED networks. The register is permanently hardwired on channel 16 and located in PPS-0 chassis.

A second status and control register is present in a 20-PP system. This register is hardwired to channel 36 and is located in PPS-1 chassis. The register is smaller and contains only the bits that affect the PPs in PPS-1. The test-error portion of the second status and control register and the one in PPS-0 may be interrogated with one test.

Channel 16 is an internal channel that is always active. The channel has a 12-bit output register to hold a descriptor word sent from a PP. The channel also has a 12-bit input register to hold the status information to be read by a PP. An output sets the channel full and keeps any other PP from outputting on the channel. An input must be made to clear the full after the output. The input frees the channel for usage by the other PPs. To maintain consistent control of this channel, all software routines that access the status and control register channel must provide an output followed by an input.

The descriptor word has 12 bits that define a word or bit address and a function code. Bits 0 through 7 contain the word or bit address that designates a 12-bit word or single bit on which the function is to be performed. Bit 8 is not used. Bits 9 through 11 contain the function code.

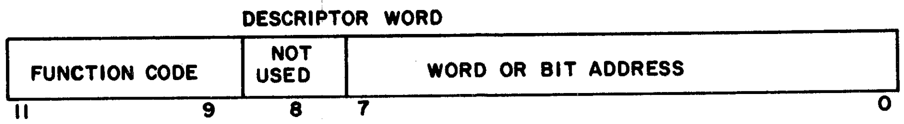


Table 5-10 lists the eight function codes designated by the function code bits.

The status bits of the status and control register receive inputs from various parts of the computer. The bits may be read from light modules (described in Peripheral Processor Subsystem Chassis in section 3) on the PPS chassis or interpreted from a program-controlled display at the display console. External status inputs always override the functions designated by function codes 0 through 7.

In some cases, groups of status bits are locked in by an error flag. For example, a SECDED-error flag locks in eight syndrome bits and six address information bits. These status bits are held until the SECDED-error flag is cleared, thereby holding the information until it is interrogated. When the error flag is cleared, the associated status bits unlock but do not necessarily clear. Also, design restrictions prevent the software from performing individual bit functions on the bits that are held by an error flag bit. For example, an individual syndrome bit cannot be tested, set, or cleared. These status bits can only be obtained by a read function (0xxx).

TABLE 5-10. DESCRIPTOR WORD FUNCTION CODES

Function Code	Function	Function Description
0	Read	The read function reads one of the 16 12-bit words specified by translations 0 through 20 (octal)
1	Test	The test function checks a bit specified by translations 0 through 277 (octal) and sends the PP a status of 1 or 0 if the bit is set or clear, respectively. The status bit is located in the bit 0 position in the 12-bit status word. The 11 other bits in the status word are 0.
2	Clear	The clear function forces a bit specified by translations 0 through 277 (octal) to 0.
3	Test/clear	The test/clear function first reads the selected bit and then clears the bit.
4	Set	The set function forces a bit specified by translations 0 through 277 (octal) to a 1.
5	Test/set	The test/set function first reads the selected bit and then sets the bit.
6	Clear all	The clear all function clears all status and control register bits except the bits indicated by program function code R in Table 5-11.
7	Test error	The test error function performs a logical OR test of the status and control register bits 0 through 39. If any bit is set, a one is returned to the PP. This allows a software routine to determine, with this single test, whether or not an error has been recorded in the status and control register. Further interrogation can be done to determine the actual error status.

The control bits of the status and control register have outputs which enable various conditions in the computer. Some of the bits may be visually read from the PPS light modules and interpreted from the display console. All control bits must be individually set with a set function because a provision does not exist for writing 12-bit words into the register.

Programming considerations for the status and control register, channel 16 (and 36), are as follows:

Instruction	Description
AJM 64	Not needed because the channel is always active
IJM 65	Not needed because the channel is always active
IAM 71	Hangs the PP with channel empty if more than one word is input
OAM 73	Hangs the PP with channel full if more than one word is output
ACN 74	Hangs the PP because the channel is always active
DCN 75	Executes, but does not disconnect the channel; becomes a two-trip pass
FAN 76	Hangs the PP because the channel is active
FNC 77	Hangs the PP because the channel is active

Table 5-11 summarizes status and control register information. The following list describes the columns in Table 5-11.

Column	Information
Word No.	Register word listed in octal (8)
Bit No.	Register bit listed in decimal (10) and octal (8)
S/C	Status (S) bits have inputs from various sources in the computer. Control (C) bits have outputs which enable various conditions in the computer.
PRGM FCTN	Indicates which programming functions (PRGM FCTN) are applicable to the status and control register bits and which of the bits are cleared at deadstart. The programming functions are indicated by abbreviations in four categories as follows:

TE	Read, test, clear, test/clear, set, test/set, clear all, and test error. This status bit is included in test error.
R	Read. No other operations can be performed.
D	Read, test, clear, test/clear, set, test/set, and clear all. This control bit is cleared at deadstart.
No abbreviation	Read, test, clear, test/clear, set, test/set, and clear all

Chan 36	Included in PPS-1
Display	Indication by light-emitting diodes (LED) located on light modules in PPS chassis

TABLE 5-11. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	M172/ 173/174	M175	S/C	PRGM FCTN	Chan 36	Dis-play	Remarks
	0	0	Read pyramid parity error	X	X	S	TE	X	X	
	1	1	CSU-0 address parity error	X	X	S	TE		X	
	2	2	CSU-1 address parity error	X	X	S	TE		X	
	3	3	SECDED error	X	X	S	TE		X	Loads and blocks bits 40 through 53
	4	4	Not used							For future enhancement
	5	5	CMC parity error	X	X	S	TE		X	Loads and locks bits 54, 55, and 139
0	6	6	PE on data received from external channel	X	X	S	TE	X	X	
	7	7	PE on data transmitted from external PP	X	X	S	TE	X	X	
	8	10	CSU-0 fault	X	X	S	TE		X	
	9	11	CSU-1 fault	X	X	S	TE		X	
	10	12	Error in second PPS	X	X	S	TE		X	Tests 0 through 39 of PPS-1
	11	13	ECS error	X	X	S	TE		X	Loads and locks bits 136 through 138
	12	14	CP-0 P register parity error	X	X	S	TE	X	X	
	13	15	CP-1 P register parity error	X		S	TE	X	X	Used only in model 174
	14	16	PP0 memory parity error	X	X	S	TE	X	X	
	15	17	PP1 memory parity error	X	X	S	TE	X	X	
	16	20	PP2 memory parity error	X	X	S	TE	X	X	
1	17	21	PP3 memory parity error	X	X	S	TE	X	X	
	18	22	PP4 memory parity error	X	X	S	TE	X	X	
	19	23	PP5 memory parity error	X	X	S	TE	X	X	
	20	24	PP6 memory parity error	X	X	S	TE	X	X	
	21	25	PP7 memory parity error	X	X	S	TE	X	X	
	22	26	PP8 memory parity error	X	X	S	TE	X	X	
	23	27	PP9 memory parity error	X	X	S	TE	X	X	
	24	30	Channel 0 parity error	X	X	S	TE	X	X	
	25	31	Channel 1 parity error	X	X	S	TE	X	X	
	26	32	Channel 2 parity error	X	X	S	TE	X	X	
	27	33	Channel 3 parity error	X	X	S	TE	X	X	
	28	34	Channel 4 parity error	X	X	S	TE	X	X	
2	29	35	Channel 5 parity error	X	X	S	TE	X	X	For channel 36, channel numbers 20 through 33 (octal) apply
	30	36	Channel 6 parity error	X	X	S	TE	X	X	
	31	37	Channel 7 parity error	X	X	S	TE	X	X	
	32	40	Channel 10 parity error	X	X	S	TE	X	X	
	33	41	Channel 11 parity error	X	X	S	TE	X	X	
	34	42	Channel 12 parity error	X	X	S	TE	X	X	
	35	43	Channel 13 parity error	X	X	S	TE	X	X	

TABLE 5-11. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS (Cont'd)

Word No. (8)	Bit No.		Description	M172/ 173/174	M175	S/C	PRGM FCTN	Chan 36	Dis- play	Remarks
	(10)	(8)								
3	36	44	Mains power failure	X	X	S	TE		X	Power/environmental abnormal condition
	37	45	Shutdown imminent	X	X	S	TE		X	
	38	46	Not used				TE	X		For future enhancement
	39	47	Not used				TE	X		
	40	50	Syndrome bit 0	X	X	S	R		X	Loaded and locked by bit 3 (memory SECDED error)
	41	51	Syndrome bit 1	X	X	S	R		X	
	42	52	Syndrome bit 2	X	X	S	R		X	
	43	53	Syndrome bit 3	X	X	S	R		X	
	44	54	Syndrome bit 4	X	X	S	R		X	
	45	55	Syndrome bit 5	X	X	S	R		X	
	46	56	Syndrome bit 6	X	X	S	R		X	
	47	57	Syndrome bit 7	X	X	S	R		X	
4	48	60	Syndrome address bit 0	X	X	S	R		X	Loaded and locked by bit 3
	49	61	Syndrome address bit 1	X	X	S	R		X	
	50	62	Syndrome address bit 2	X	X	S	R		X	
	51	63	Syndrome address bit 16	X	X	S	R		X	
	52	64	Syndrome address bit 17	X	X	S	R		X	
	53	65	Syndrome address bit 3	X	X	S	R		X	
	54	66	Parity error port code bit 0	X	X	S	R		X	From CMC. Identifies port. Loaded and locked by bit 5
	55	67	Parity error port code bit 1	X	X	S	R		X	
	56	70	Breakpoint port code bit 0	X	X	S	R		X	Loaded and locked by bit 77
	57	71	Breakpoint port code bit 1	X	X	S	R		X	
	58	72	Breakpoint function code bit 0	X	X	S	R		X	
	59	73	Breakpoint function code bit 1	X	X	S	R		X	
5	60	74	PPS P register bit 0	X	X	S	R	X	X	If bit 83 is clear, bits 60 through 71 display P of the PP selected by bits 120 through 123, and bits 72 through 75 display selected PP. If bit 83 is set, the content of P register is latched and retained on every CM breakpoint bit. If bit 76 sets when bit 83 is set, bits 60 through 75 are held until bit 76 is cleared. Refer to text for more detailed information.
	61	75	PPS P register bit 1	X	X	S	R	X	X	
	62	76	PPS P register bit 2	X	X	S	R	X	X	
	63	77	PPS P register bit 3	X	X	S	R	X	X	
	64	100	PPS P register bit 4	X	X	S	R	X	X	
	65	101	PPS P register bit 5	X	X	S	R	X	X	
	66	102	PPS P register bit 6	X	X	S	R	X	X	
	67	103	PPS P register bit 7	X	X	S	R	X	X	
	68	104	PPS P register bit 8	X	X	S	R	X	X	
	69	105	PPS P register bit 9	X	X	S	R	X	X	
	70	106	PPS P register bit 10	X	X	S	R	X	X	
	71	107	PPS P register bit 11	X	X	S	R	X	X	

TABLE 5-11. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS (Cont'd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	M172/ 173/174	M175	S/C	PRGM FCTN	Chan 36	Dis-play	Remarks
6	72	110	PP code bit 0	X	X	S	R	X	X	Refer to remarks for bits 60 through 71
	73	111	PP code bit 1	X	X	S	R	X	X	
	74	112	PP code bit 2	X	X	S	R	X	X	
	75	113	PP code bit 3	X	X	S	R	X	X	
	76	114	PPS breakpoint bit	X	X	S		X	X	Loads and locks bits 56 through 59
	77	115	CMC breakpoint match	X	X	S		X	X	
	78	116	Clear central memory busy	X	X	C		X		Clear busy FF in PPS
	79	117	Set C5 full	X	X	C		X		One-shot operation
	80	120	Force zero parity on channels	X	X	C	D	X		One-shot operation
	81	121	Force zero parity on PPM	X	X	C	D	X		
	82	122	Not used							For future enhancement
	83	123	PPS breakpoint mode select	X	X	C	D	X		Refer to remarks for bits 60 through 75
7	84	124	All PPs 500-nsec major cycle	X	X	C	D		X	Controls PPS-0 and PPS-1
	85	125	Inhibit PPS request to CMC	X	X	C	D	X	X	
	86	126	Not used						X	For future enhancement
	87	127	Not used						X	
	88	130	Not used						X	
	89	131	Not used						X	
	90	132	Not used						X	
	91	133	Not used						X	
	92	134	Not used						X	
	93	135	Not used						X	
	94	136	Not used						X	
	95	137	Stop on PPM parity error	X	X	C	D	X	X	Applies to all PPs
10	96	140	Breakpoint address bit 0	X	X	C				Absolute 18-bit address bits 96 through 113 are sent to and used by CMC to establish breakpoint address when bits 116 and/or 117 are set
	97	141	Breakpoint address bit 1	X	X	C				
	98	142	Breakpoint address bit 2	X	X	C				
	99	143	Breakpoint address bit 3	X	X	C				
	100	144	Breakpoint address bit 4	X	X	C				
	101	145	Breakpoint address bit 5	X	X	C				
	102	146	Breakpoint address bit 6	X	X	C				
	103	147	Breakpoint address bit 7	X	X	C				
	104	150	Breakpoint address bit 8	X	X	C				
	105	151	Breakpoint address bit 9	X	X	C				
	106	152	Breakpoint address bit 10	X	X	C				
	107	153	Breakpoint address bit 11	X	X	C				

TABLE 5-11. STATUS AND CONTROL REGISTER BIT ASSIGNMENT (Cont'd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	M172/ 173/174	M175	S/C	PRGM FCTN	Chan 36	Dis-play	Remarks
11	108	154	Breakpoint address bit 12	X	X	C				
	109	155	Breakpoint address bit 13	X	X	C				
	110	156	Breakpoint address bit 14	X	X	C				
	111	157	Breakpoint address bit 15	X	X	C				
	112	160	Breakpoint address bit 16	X	X	C				
	113	161	Breakpoint address bit 17	X	X	C				
	114	162	Breakpoint condition code bit 18	X	X	C				Select function RD/ WT/RNI or all three to CMC for port selection
	115	163	Breakpoint condition code bit 19	X	X	C				
	116	164	Breakpoint condition code bit 20	X	X	C				
	117	165	Breakpoint condition code bit 21	X	X	C				
	118	166	Inhibit single error report	X	X	C			X	Single errors are not recorded in SCR when set
	119	167	Not used							For future enhancement
12	120	170	PP select code bit 0	X	X	C	D	X	X	Select 1 of 10 PPs for forced exit, deadstart, or display
	121	171	PP select code bit 1	X	X	C	D	X	X	
	122	172	PP select code bit 2	X	X	C	D	X	X	
	123	173	PP select code bit 3	X	X	C	D	X	X	
	124	174	PP select auto/manual mode	X	X	C	D	X	X	Clear = manual
	125	175	Force exit on selected PP	X	X	C	D	X		One-shot operation
	126	176	Force PP deadstart on selected PP	X	X	C	D	X		Set forces deadstart. PP remains in deadstart condition until bit is cleared.
	127	177	CSU, CMC, CPU master clear	X	X	C	D			
	128	200	Force zero SECDED code and parity CMC to CM	X	X	C				
	129	201	Force zero address parity CMC to CM	X	X	C				
	130	202	Disable address parity error	X	X	C				
	131	203	Not used							For future enhancement
13	132	204	Force zero parity code 0	X	X	C				ECS coupler
	133	205	Force zero parity code 1	X	X	C				ECS coupler
	134	206	Refresh margin slow	X	X	C				
	135	207	Refresh margin fast	X	X	C				
	136	210	ECS transfer error code 0	X	X	S	R			Bits 136 through 138 are loaded and locked by bit 11.
	137	211	ECS transfer error code 1	X	X	S	R			
	138	212	ECS transfer error code 2	X	X	S	R			
	139	213	CMC adrs/data parity error	X	X	S	R		X	Loaded and locked by bit 5. Clear = data error
	140	214	Not used							
	141	215	Clock frequency magnitude 0	X	X	C	D			Bits 141 through 143 are code bits for selecting clock margins.
	142	216	Clock frequency magnitude 1	X	X	C	D			
	143	217	Clock frequency slow/fast	X	X	C	D			Clear=slow

TABLE 5-11. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS (Cont'd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	M172/ 173/174	M175	S/C	PRGM FCTN	Chan 36	Dis-play	Remarks
14	144	220	RVM address bit 0 status		X	S			X	Indicates module having reference voltage margins (RVM) applied
	145	221	RVM address bit 1 status		X	S			X	
	146	222	RVM address bit 2 status		X	S			X	
	147	223	RVM address bit 3 status		X	S			X	
	148	224	RVM address bit 4 status		X	S			X	
	149	225	RVM address bit 5 status		X	S			X	
	150	226	RVM hi/lo		X	S			X	Clear=lo
	151	227	RVM all/one		X	S			X	Clear=one
	152	230	Clock pulse width narrow		X	C			X	
	153	231	Clock pulse width wide		X	C			X	
	154	232	Select hi/lo RVM		X	C				Clear=lo
	155	233	Select all/one RVM		X	C				Clear=one (refer to text
15	156	234	RVM quadrant 0 select		X	C				Used with bits 154 and 155
	157	235	RVM quadrant 1 select		X	C				
	158	236	RVM quadrant 2 select		X	C				
	159	237	RVM quadrant 3 select		X	C				
	160	240	RVM quadrant 4 select		X	C				
	161	241	RVM quadrant 5 select		X	C				
	162	242	RVM quadrant 6 select		X	C				
	163	243	RVM quadrant 7 select		X	C				
	164	244	RVM quadrant 8 select		X	C				
	165	245	RVM quadrant 9 select		X	C				
	166	246	RVM quadrant 10 select		X	C				
	167	247	RVM quadrant 11 select		X	C				
16	168	250	RVM module address bit 0		X	C				
	169	251	RVM module address bit 1		X	C				
	170	252	RVM module address bit 2		X	C				
	171	253	RVM module address bit 3		X	C				
	172	254	RVM module address bit 4		X	C				
	173	255	RVM module address bit 5		X	C				
	174	256	PPS to CMC zero address parity	X	X	C		X		
	175	257	PPS to CMC zero data parity	X	X	C		X		
	176	260	Not used							For future enhancement
	177	261	Not used							
	178	262	Not used							
	179	263	Not used							

TABLE 5-11. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS (Cont'd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	M172/ 173/174	M175	S/C	PRGM FCTN	Chan 36	Dis-play	Remarks
17	180	264	Not used							For future enhancement
	181	265	Not used							
	182	266	Not used							
	183	267	Double error	X	X	S			X	
	184	270	CP-0 to CMC zero address parity	X		C				
	185	271	CP-1 to CMC zero address parity	X		C				Used only in model 174
	186	272	CP-0 to CMC zero data parity	X		C				
	187	273	CP-1 to CMC zero data parity	X		C				Used only in model 174
	188	274	Software flag 0	X	X	C		X		Diagnostic aids
	189	275	Software flag 1	X	X	C		X		
	190	276	Not used							For future enhancement
	191	277	Not used							
20	192	300	CP-0 stopped	X	X	S	R		X	
	193	301	CP-1 stopped	X		S	R		X	Used only in model 174
	194	302	ECS in progress flag	X	X	S	R		X	
	195	303	Monitor flag CP-0	X	X	S	R		X	
	196	304	Monitor flag CP-1	X		S	R		X	Used only in model 174
	197	305	PPM reconfiguration bit 0	X	X	S	R			
	198	306	PPM reconfiguration bit 1	X	X	S	R			
	199	307	PPM reconfiguration bit 2	X	X	S	R			
	200	310	PPM reconfiguration bit 3	X	X	S	R			
	201	311	PPM reconfiguration bit 4	X	X	S	R			PPS select
	202	312	Not used							For future enhancement
	203	313	Not used							

DATA CHANNEL CONVERTERS

The following programming information is for one data channel converter (DCC) and applies to each DCC in a basic or expanded CDC CYBER 170 System.

CODES

Two sets of codes are required to operate a 3000 Series peripheral equipment through a DCC.

- Function and status response codes for the DCC
- Connect, function, and status codes for the specific 3000 Series equipment

The DCC function codes allow the computer system to connect to the 3000 Series equipment and to transmit 3000 Series function codes to the connected equipment. Function codes also permit the sensing of both DCC and external equipment status and enable the flow of data between the data channel and the 3000 Series equipment through the DCC.

The 3000 Series codes include connect, function, and status reply. These codes prepare a connected equipment for an I/O operation. They do not affect unconnected equipment. The 3000 Series status codes monitor the operating conditions of several pieces of equipment (refer to 3000 Series equipment manuals for a complete list of these codes).

Function codes are transmitted to the DCC by PP function A on channel d (FAN, 76) and function m on channel d (FNC, 77) instructions. Bit 0 is rightmost in all codes.

The function codes are:

Function	Code
Select DCC	2000 ①
Deselect DCC	2100
Connect equipment - Mode I	NUUU ②
Connect equipment - Mode II	1000
Function transmit - Mode I	0FFF ③
Function initiate - Mode II	1100
Input EOR initiate	14XX ④
Input initiate	15XX
Output initiate	16XX
Deactivate option	XX4X/XX6X
Function master clear	1700
DCC status request	1200
Equipment status request	1300

Notes:

① One DCC is assigned different select and deselect codes, such as 2200 and 2300 or 2400 and 2500, when two DCCs share a common data channel.
② N equals 4 through 7 (equipment number), and UUU equals lower 9 bits of connect code.
③ FFF equals lower 9 bits of function code.
④ Initiate conditions are defined by XX.

In the following code descriptions, some of the code characters have been expanded to show the character bits. For example, the 14XX code is expanded to 14Xoox, where oox represents the last three character bits. The XXX1 code is expanded to XXXoo1, where oo1 represents the last three character bits.

Function Codes

Select Converter (2000)

Function code 2000 selects the DCC from among the equipment sharing the same data channel. One DCC is assigned different select and deselect codes, such as 2200 and 2300 or 2400 and 2500, when two DCCs share a common data channel. A deadstart master clear automatically selects all DCCs in the computer system.

Deselect Converter (2100)

Function code 2100 deselects the DCC. The DCC must be deselected before other equipment on the same data channel is used.

Connect Equipment, Mode 1 (NUUU)

Function code NUUU connects 3000 Series equipment 4, 5, 6, or 7 and units UUU, where N equals the equipment number (4 through 7) and UUU (lower 9 bits) equals the unit number.

Connect Initiate, Mode II (1000)

Function code 1000, specifying a mode II operation, causes the DCC to send the next data word received to the 3000 Series equipment as a connect code. Code 1000 connects 3000 Series equipment 0 through 7. The 1000 function code must be followed by a one-word data output. The data is the connect code.

Function Transmit, Mode 1 (0FFF)

Function code 0FFF, specifying a mode I operation, causes the DCC to transmit the 12-bit function code (0FFF) to the connected 3000 Series equipment. FFF can be the lower 9 bits of any 12-bit code whose upper 3 bits are zeros.

Function Initiate, Mode II (1100)

Function code 1100, specifying a mode II operation, causes the DCC to send the next data word received to the connected 3000 Series equipment as a function code. This code can be used to transmit any 3000 Series function code to the connected equipment. The 1100 code should be followed by a one-word data output. The data is the function code.

Input EOR Initiate (14Xoox)

Function code 14Xoox prepares the DCC for an input operation. The code terminates the input by either an end-of-record signal from the 3000 Series equipment or by a channel disconnect from the PP. Initiate conditions are defined by XX.

Input Initiate (15Xoox)

Function code 15Xoox prepares the DCC for an input operation. The code terminates the input by a channel disconnect only. (Refer to Input EOR Initiate description in this section.) A negate BCD conversion line is enabled to the external equipment when bit 0 of code 15Xoox is set. The negate BCD conversion remains in effect until a 14X0, 15X0, or 16X0 function code is received.

The 15Xoox function code should not be used for magnetic tape units. A magnetic tape transport stops tape motion when it senses the end of a record. However, when code 15XX is in effect, the DCC does not disconnect the data channel on the end of a record. If the specified word matches the record word count, the PP exits the IAM instruction with the channel active.

Output Initiate (16XX)

Function code 16XX prepares the DCC for an output operation. The code terminates the output by a channel disconnect. (Refer to Input EOR Initiate description in this section.) A negate BCD conversion line is enabled to the external equipment when bit 0 of code 16XX is set. The negate BCD conversion remains in effect until a 14X0, 15X0, or 16X0 function code is received.

Deactivate Option (XX6X) and (XX4X)

Function codes XX6X and XX4X allow two additional methods of generating an inactive signal in the DCC during a read or write operation.

- The XX6X code must be sent to the DCC, with an input or output function code (1460, 1560, or 1660). This sends an active signal to the data channel when this option is selected in the DCC, and an interrupt-override signal is returned from the peripheral controller. The XX6X code may be used for any 3000 Series peripheral controller that has an interrupt-override signal feature.

  The interrupt-override signal is generated in a 3000 Series peripheral controller when interrupt on abnormal end-of-operation is selected and an abnormal condition exists. The interrupt-override signal is returned to the DCC where an inactive signal is generated and sent to the data channel.

- The XX4X code must be sent to the DCC with input or output function code 1440, 1540, or 1640. This code is used for 3000 Series peripheral controllers that do not have the interrupt-override signal feature.

  When an abnormal end-of-operation is selected in the 3000 Series peripheral controller and an abnormal condition exists, an abnormal end-of-operation status code 1XXX is returned to the DCC. The DCC senses for status code 1XXX and generates an inactive signal that is sent to the data channel.

Function Master Clear (1700)

Function code 1700 master clears all 3000 Series equipment attached to the DCC, as well as all the conditions within the DCC.

Data Channel Converter Status Request (1200)

Function code 1200 permits the PP to input DCC status. A one-word input must follow to read the status response.

Equipment Status Request (1300)

Function code 1300 permits the PP to input the status response from the connected 3000 Series equipment. A one-word input must follow to read the status word.

NOTE

Any 1XXX function code sent to the DCC clears the previous 1XXX function condition.

Status Reply Codes

Two types of status codes are available from the DCC, DCC status codes and equipment status codes.

Function code 1200 makes the DCC status response available to the PP. A one-word data input must follow to read the status word. The 12-bit DCC status responses are:

Code	Description
XXX0	Reply
XXXxx1	Reject (internal or external)
XXXx11	Internal reject
XXXX1xx	Transmission parity error between DCC and 3000 Series peripheral controller
XX1X-2XXX	Equipment interrupts
1xxXX1xx	Transmission parity error on data from PP to DCC

Each piece of 3000 Series peripheral equipment provides a 12-bit status response. The response code is available at the time the equipment is connected to the DCC or after the peripheral equipment rejects a connect code. Each bit in the response code indicates a condition within the peripheral equipment, such as ready, busy, or end-of-tape. A PP makes a status request to the connected 3000 Series equipment by sending a 1300 function code to the DCC. The PP then makes a one-word input to read the response.

Equipment status codes differ for each equipment. The codes are listed in the manual describing the individual equipment. The DCC status codes are defined in the following paragraphs.

Reply (XXX0)

Bits 0 through 2 clear when the 3000 Series equipment returns a reply signal to the DCC in response to a connect or function code.

Reject (Internal or External) (XXXxx1)

Bit 0 sets when the 3000 Series equipment returns a reject signal to the DCC in response to a connect or function code. An internal reject signal sets both bits 0 and 1.

Internal Reject (XXXx11)

Bits 0 and 1 set, after a 100-microsecond delay, if the 3000 Series equipment fails to return a reply or a reject signal to the DCC in response to a connect or function code.

Transmission Parity Error (XXX1xx)

Bit 2 sets when a parity error occurs on a function code or data transfer between the DCC and the 3000 Series equipment. A parity error on a connect code does not set bit 2.

Equipment Interrupts (XX1X-2XXX)

One of bits 3 through 10 indicates the interrupt signal from one of eight possible 3000 Series pieces of equipment. If equipment N sends an interrupt, status bit N plus 3 is set and remains set until the equipment drops the interrupt signal.

Transmission Parity Error on Data From Channel (1xxXX1xx)

A parity error was detected on data transmitted from the data channel. The DCC retransmits this data with incorrect parity and sets bits 2 and 11.

SELECTING THE DATA CHANNEL CONVERTER

The DCC must be selected from among the other equipment that shares the same data channel before it communicates with 3000 Series peripheral equipment. The selected (2000) function code, transmitted by a PP FAN (76) or FNC (77) instruction, selects the DCC. One DCC is assigned different select and deselect codes, such as 2200 and 2300 or 2400 and 2500 when two DCCs share a common data channel. Selection activates the DCC and renders inactive all other I/O equipment on the data channel.

A deadstart master clear automatically selects all DCCs in the computer system.

DESELECTING THE DATA CHANNEL CONVERTER

Once selected, the DCC remains selected until it is deselected by function code 2100. The DCC must always be deselected before any other I/O equipment on the same data channel can be used.

If two DCCs on the same data channel have been selected by a deadstart master clear, the first DCC must be deselected before the second DCC can be deselected because a selected DCC does not relay codes sent from the data channel to more distant equipment.

CONNECTING TO 3000 SERIES EQUIPMENT

One of eight possible 3000 Series controllers attached to the DCC may be connected after the DCC is selected. The connect operation activates one controller and automatically deactivates the other seven controllers so that only one of eight possible controllers can be connected at a given time.

A 12-bit connect code (Figure 5-11) connects a 3000 Series controller to a DCC.

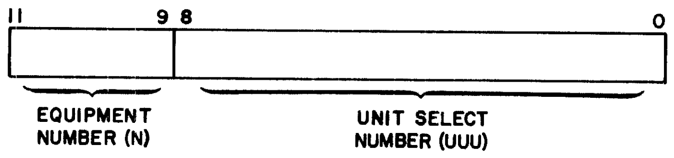


Figure 5-11. Connect Code Format

Bits 9 through 11 indicate the equipment number of the equipment to be connected. Each piece of 3000 Series equipment is assigned a number (0 through 7) by an eight-position equipment number switch. Bits 0 through 8 designate one of several possible units which are subordinate to the equipment. For example, a tape controller ranks as a piece of equipment. Each attached tape transport is a unit designated by a unit select number. Bits 0 through 8 are not used with a piece of equipment that has no subordinate units, such as a card reader.

A connect code is sent from a PP, through the DCC, to an attached 3000 Series controller. Methods of sending a connect code are mode I connect and mode II connect. A mode I connect operation requires only one DCC function code from the PP, but is restricted to connecting only equipment numbered 4 through 7. A mode II connect operation requires a DCC function code followed by a one-word data output. Mode II can connect any of the eight possible pieces of equipment numbered 0 through 7.

A connect is broken only by connecting to another piece of equipment through a deadstart master clear or a DCC function master clear (1700). Deselecting the DCC or disconnecting the data channel does not clear a connect.

Mode I Connect

The DCC performs a mode I connect operation whenever the PP sends a function code in the form 4UUU through 7UUU. The DCC forwards the function code to the attached 3000 Series equipment as a connect code. Normally, the equipment corresponding to the upper octal digit (4 through 7) connects, and any previously connected equipment automatically disconnects.

If any equipment connects successfully, it returns a reply signal to the DCC which sends an inactive signal to the data channel. The reply signal disconnects the data channel, making it available for another operation.

Some 3000 Series equipment may not be able to connect under certain conditions. In such cases, the equipment returns a reject signal to the DCC. The reject signal acts as a reply, causing the DCC to send an inactive signal to the data channel. In addition, the reject signal sets status bit 0 in the DCC, indicating that the connect code was rejected. The conditions which cause the 3000 Series equipment to reject a connect code are listed in the reference manual for each equipment. Neither a reply nor a reject signal is returned to the DCC if a connect code addresses a nonexistent equipment or if a malfunction occurs in the equipment. In such cases, the DCC generates an internal reject signal after a 100-microsecond delay. An internal reject signal causes the DCC to send an inactive signal to the data channel. The internal reject signal also sets reject status bit 0 and internal reject status bit 1 in the DCC.

The 3000 Series equipment checks each connect code sent from the DCC for parity. If a parity error occurs, no equipment connects and neither a reply nor a reject signal is returned to the DCC. The DCC generates an internal reject signal after a 100-microsecond delay.

Mode II Connect

A mode II connect operation requires a function code and a one-word output to the DCC.

- A connect initiate (1000) function code is sent to the DCC by FAN (76) or FNC (77) instruction. This code conditions the DCC for a mode II connect operation. Function code 1000 is not sent to the 3000 Series equipment. The DCC returns an inactive signal to release the data channel.
- A one-word output containing the connect code is sent to the DCC by an output A words from m on channel d (OAM, 73) or an output from A on channel d (OAN, 72) instruction. The DCC forwards this output word to the 3000 Series equipment as a connect code.

The possible responses to the connect code are:

• Reply	Indicates addressed equipment successfully connected.
• Reject	Indicates addressed equipment could not connect. Reject status bit 0 is set in the DCC.
• No response	The DCC generates an internal reject signal after a 100-microsecond delay. Internal reject status bits 1 and 0 are both set.

Any of the above three responses causes the DCC to send an empty signal to the data channel, indicating receipt of the output word. A jump to m if channel d full (FJM, 66) instruction should follow the data output to delay the program until the DCC accepts the output word if an OAN (72) instruction has been executed. A disconnect channel d (DCN, 75) instruction should follow to deactivate the data channel.

The 3000 Series equipment checks each connect code sent from the DCC for parity, identical to mode I connect operation. If a parity error occurs, no equipment connects, and neither a reply nor a reject signal is returned to the DCC. The DCC generates an internal reject signal after a 100-microsecond delay.

Check the DCC status response for a reject after a mode II connect operation is complete.

NOTE

A status check should follow only after the mode II connect operation is complete. There is no response from the 3000 Series equipment when the connect initiate code (1000) is sent to the DCC. Thus, a status check at this time is not significant.

SENDING FUNCTION CODES TO 3000 SERIES EQUIPMENT

A piece of 3000 Series equipment accepts 12-bit function codes from the DCC after it is connected. Function codes establish operating conditions within an equipment or initiate operations, such as tape rewind. The function codes applicable to the 3000 Series equipment are listed in the reference manual for each equipment.

The function codes sent from the DCC to the 3000 Series equipment are distinct from function codes transmitted from the PP to the DCC.

Two methods are used to transmit function codes to a 3000 Series equipment.

Mode I	A mode I function operation requires only a single PP function instruction (FAN or FNC), but is restricted to a nine-bit function code.
Mode II	A mode II function operation requires a function instruction followed by a one-word data output. A full 12-bit function code can be sent to the 3000 Series equipment by mode II procedures.

Mode I Function

A mode I function operation is similar to a mode I connect operation. The DCC performs a mode I function operation whenever a PP sends a 0FFF function code to the DCC. FFF can be any nine-bit 3000 Series function code. The DCC forwards the 0FFF to the connected equipment as a function code.

The DCC receives one of three possible responses to a function code from the 3000 Series equipment.

• Reply	Indicates that the 3000 Series equipment accepted the code.
• Reject	Indicates that the 3000 Series equipment did not accept the code. Reject status bit 0 is set in the DCC.
• No response	If, after a 100-microsecond delay, neither a reply nor a reject signal is received, the DCC generates an internal reject signal that sets internal reject status bits 0 and 1.

A status check should follow a mode I function operation to test for a reject signal or a parity error at the 3000 Series peripheral controller.

Mode II Function

A mode II function operation is similar to a mode II connect operation requiring a function code and a one-word output to the DCC.

- A function initiate (1100) function code is sent to the DCC by a FAN (76) or FNC (77) instruction. This code conditions the DCC for a mode II function operation and is not forwarded to the 3000 Series equipment. The DCC returns an inactive signal to release the data channel.
- A one-word output containing the desired 12-bit function code is sent to the DCC by an OAN (72) or an OAM (73) instruction. The DCC forwards this output word to the 3000 Series equipment as a function code.

The responses to a mode II function operation are the same as for a mode I function operation.

● Reply	Indicates that the 3000 Series equipment accepted the code.
● Reject	Indicates that the 3000 Series equipment did not accept the code. Reject status bit 0 is set in the DCC.
● No response	If, after a 100-microsecond delay, neither a reply nor a reject signal is received, the DCC generates an internal reject signal that sets internal reject status bits 0 and 1.

Any of the above three responses causes the DCC to send an empty signal to the data channel, indicating receipt of the output word. A full FJM (66) instruction should follow the data output to delay the program until the DCC accepts the output word if an OAN (72) instruction has been executed. A DCN (75) instruction should follow to deactivate the data channel.

A status check should follow a mode II function operation to test for a reject signal or a parity error at the 3000 Series peripheral controller.

DATA TRANSFER

An input or output operation can proceed only after the DCC is selected and the desired equipment is connected to the DCC.

Input Operation

An input operation requires the following actions.

- Send an input initiate or an input EOR initiate function code to the DCC. This code conditions the DCC for an input operation.
- Execute an activate channel instruction for the data channel. This signals the equipment, through the DCC, to begin sending data. For example, it starts tape or card motion.
- Execute an input instruction to read the data from the sending device.

The basic input function codes are 14XX and 15XX. Each code prepares the DCC for a specific input operation.

Input function code 1400 terminates the input operation either when the 3000 Series peripheral equipment reaches an end-of-record or when a data channel disconnect is received from the PP. An end-of-record sensed by the 3000 Series equipment causes the DCC to send an inactive signal which disconnects the data channel. The PP then exits to the next instruction.

Input function code 1401 suppresses the internal-to-external BCD conversion that normally takes place in some 3000 Series equipment. Code 1401 is identical to code 1400 in other respects.

On some 3000 Series equipment, a significant delay occurs between the channel activate instruction that signals the start of an input operation and the time that the first data word is available from the equipment. For example, in the 3248 Card Read Controller, a 20-millisecond delay occurs between the start of card motion and the availability of the first card column. During this period, the PP can perform another task. The latent period is different for each 3000 Series equipment. Its length can be found in the reference manual describing the device. An input instruction should immediately follow the channel activate instruction if the delay is unknown.

The DCC does not deactivate the data channel on end-of-record if input initiate code 15XX is used. A channel disconnect instruction must immediately follow the input instruction to notify the equipment of the end of operation.

Input function codes 14XX and 15XX remain in effect until the next DCC function code is received. The negate internal-to-external BCD condition, established by codes 14X1 and 15X1, is cleared when the PP sends a new I/O function with bit 0 clear. An input operation is followed normally by a status request function code. Thus, each input operation usually requires a new input initiate code.

Output Operation

An output operation requires the following actions.

- Send an output initiate function code to the DCC to prepare it for an output operation.
- Execute an activate channel instruction for the data channel. This signals the equipment, through the DCC, that an output operation is about to begin. The connected device prepares to receive data. For example, the device starts card or tape motion.
- Execute an output instruction to send data to the 3000 Series equipment.
- Execute a full jump to ensure that the 3000 Series equipment has accepted the last word. Execute a DCN (75) instruction. This step releases the data channel and notifies the 3000 Series equipment of the end of the record.

The output initiate code is 16XX. Output function code 1601 suppresses the internal-to-external BCD conversion that normally takes place in some 3000 Series equipment. Code 1601 is identical to code 1600 in other respects.

On some 3000 Series equipment, a delay occurs between the channel activate instruction that signals the start of an output operation and the time that the equipment is ready to accept the first data word. During this period, the PP can perform another task. An output instruction should immediately follow the channel activate instruction if the delay is unknown.

Output initiate code 1600 remains in effect until the next DCC function code is received. The negate internal-to-external BCD condition established by code 16X1 is cleared when the PP sends a new I/O function with bit 0 clear. An output operation is normally followed by a status request function code which clears the output condition in the DCC. Thus, each output operation usually requires a new output initiate code.

PARITY CHECKING

The DCC checks parity on all function codes and data received from the data channel or the connected 3000 Series controllers. The DCC generates parity for data sent in any direction.

Function Codes from PPS to DCC

The PPS transmits a 12-bit function code plus 1 parity bit to the DCC. The DCC checks each function code that it receives for odd parity. If the DCC detects a parity error, the following occurs.

- The connect or function signal to the peripheral controller is blocked.
- The DCC does not send an inactive signal to the data channel. A timeout must be executed, and if no inactive signal is received, a DCN must follow.
- Parity error status bits 2 and 11 in the data channel status word remain clear.
- The function register in the DCC is cleared. Therefore, the function is not executed.

Data from PPS to DCC

The PPS transmits a 12-bit data byte (includes functional data on mode II connect or function operation) plus 1 parity bit to the DCC. The DCC checks each data byte that it receives for odd parity. If the DCC detects a parity error, the following occurs.

- Parity error status bits 2 and 11 set in the channel status word.
- The parity bit received from the data channel is sent unchanged to the peripheral controller with the data byte. The controller also detects the parity error and responds.
- The response to a mode II functional data byte is either an external or internal reject signal.

Data from DCC to PPS

The 3000 Series controller transmits a 12-bit function code plus 1 parity bit to the DCC. The DCC checks each data byte that it receives for odd parity. If the DCC detects a parity error, the following occurs.

- Parity error status bit 2 sets in the data channel status word.
- The data byte and the parity bit received from the controller are sent unchanged to the PPS.
- Operations proceed as normal.

Status Words from DCC to PPS

There is no parity on status words sent from the peripheral controller to the DCC. The DCC transmits a 12-bit function code plus 1 parity bit to the PPS. The PPS checks each word it receives for odd parity and sets channel bit X in its status and control register if a parity error is detected.

CLEARING A PARITY ERROR

A DCC function master clear (1700) must be executed to clear a parity error condition in the 3000 Series equipment if a status check reveals that a parity error occurred. This action also clears DCC parity error status bits 2 and 11.

Each piece of equipment must complete its operation before the master clear code is issued if the DCC is alternately operating two or more pieces of 3000 Series equipment on a time-sharing basis. This procedure ensures that the master clear code does not cause a loss of data.

DISPLAY STATION

KEYBOARD

A PP must transmit a one-word function code (7020_8) to request data from the keyboard of the display station. The code prepares the display controller for an input operation. The PP then checks for an active input channel and receives one character from the keyboard. This character is entered as the lower six bits of the word. The upper bits are cleared. There is no status report by the keyboard. Table 5-12 lists the keyboard character codes.

TABLE 5-12. KEYBOARD CHARACTER CODES

Character	Code	Character	Code	Character	Code
No data	00	P	20	5	40
A	01	Q	21	6	41
B	02	R	22	7	42
C	03	S	23	8	43
D	04	T	24	9	44
E	05	U	25	+	45
F	06	V	26	-	46
G	07	W	27	*	47
H	10	X	30	/	50
I	11	Y	31	(	51
J	12	Z	32	)	52
K	13	0	33	Left blank key	53
L	14	1	34	=	54
M	15	2	35	Right blank key	55
N	16	3	36	,	56
O	17	4	37	.	57
				Carriage return	60
				Backspace	61
				Space	62

DATA DISPLAY

Data is displayed within an 8-inch by 8-inch area of a cathode-ray tube (CRT). The display can be alphanumeric (character mode) or graphic (dot mode). There are 262,144 dot locations arranged in a 512-by-512 format. Each dot position is determined by the intersection of X and Y coordinates. The lower left corner dot is address X=6000 and Y=7000, and the upper right corner dot is address X=6777 and Y=7777.

Character Mode

In character mode, large, medium, and small characters are provided. Large characters are arranged in a 32-by-32 dot format with 16 characters per line. Medium characters are arranged in a 16-by-16 dot format with 32 characters per line. Small characters are arranged in an 8-by-8 dot format with 64 characters per line. Table 5-13 lists the character codes.

TABLE 5-13. DISPLAY CHARACTER CODES

Character	Code	Character	Code	Character	Code
Space	00	P	20	5	40
A	01	Q	21	6	41
B	02	R	22	7	42
C	03	S	23	8	43
D	04	T	24	9	44
E	05	U	25	+	45
F	06	V	26	-	46
G	07	W	27	*	47
H	10	X	30	/	50
I	11	Y	31	(	51
J	12	Z	32	)	52
K	13	0	33	Space	53
L	14	1	34	=	54
M	15	2	35	Space	55
N	16	3	36	,	56
O	17	4	37	.	57

Dot Mode

In dot mode, display dots are positioned by the X and Y coordinates. The X coordinates position the dots horizontally. The Y coordinates position the dots vertically and unblank the CRT for each dot. Horizontal lines are formed by a series of X and Y coordinates. Vertical lines are formed by a single X coordinate and a series of Y coordinates.

Codes

A single function word is transmitted to select the presentation, mode, and character size (character mode only). Figure 5-12 illustrates the function word format. The word following the function word specifies the starting coordinates for the display (for either mode). Figure 5-13 illustrates coordinate data word. In character mode, the following words are display character codes. Figure 5-14 illustrates the character word.

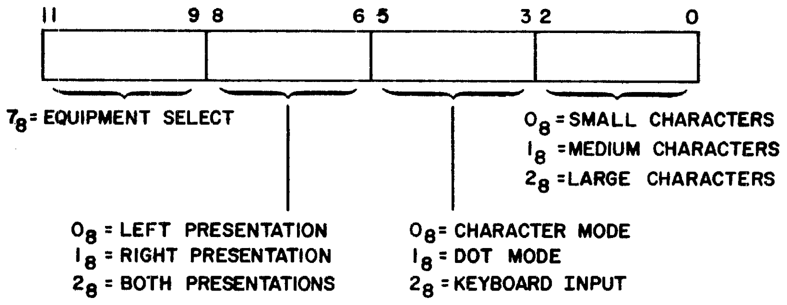
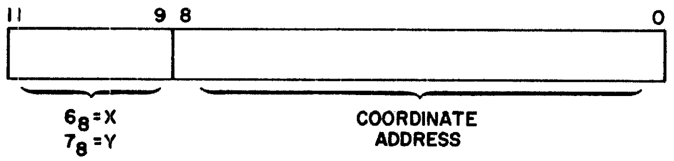


Figure 5-12. Display Station Output Function Code



NOTE:
IN DOT MODE, EACH Y COORDINATE TRANSMITTED FORCES A DOT DISPLAY.

Figure 5-13. Coordinate Data Word

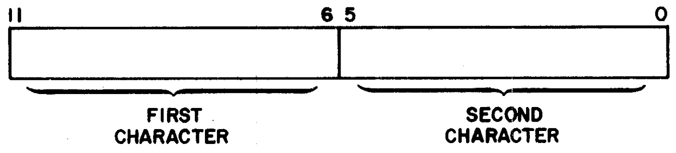


Figure 5-14. Character Data Word

When the display operation has started, the controller regulates character spacing on the line. A new coordinate data word must be sent to start each line. If new coordinates are not specified, data is written on the line specified by the active coordinate word, and information already on that line is overwritten. Character sizes can be mixed by sending a new function word and coordinate word for each size change. Spacing on a line can be varied by sending a coordinate word for the character which is to be spaced differently.

PROGRAMMING EXAMPLE

The following programming example (Figure 5-15) requests an input of one line of data from the display station and displays this data on the CRT as it is being typed.

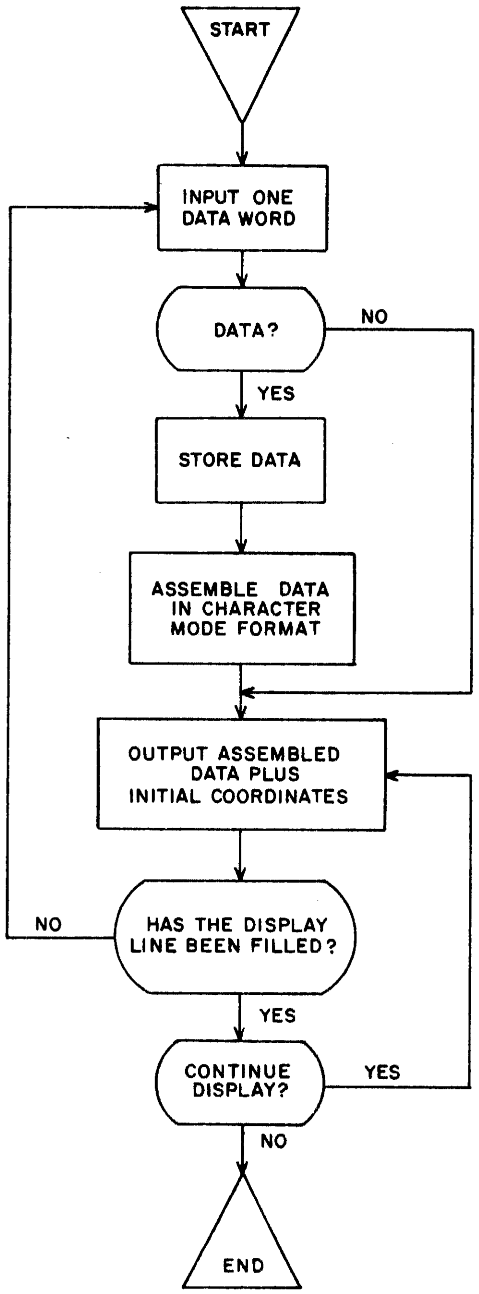

START
INPUT ONE DATA WORD
DATA?
NO
YES
STORE DATA
ASSEMBLE DATA IN CHARACTER MODE FORMAT
OUTPUT ASSEMBLED DATA PLUS INITIAL COORDINATES
NO
HAS THE DISPLAY LINE BEEN FILLED?
YES
CONTINUE DISPLAY?
YES
NO
END

Figure 5-15. Receive and Display Program Flowchart

INSTRUCTION TIMING A

CENTRAL PROCESSOR INSTRUCTION TIMING

Execution times for the central processor (CP) instructions are given in Table A-1 for models 172 through 174 and Table A-2 for model 175. The times listed in the execution time columns assume that no conflicts occur. However, a delay results unless all the conditions listed in the timing notes columns exist for the particular instruction. The timing notes columns are keyed to the notes listed at the end of the tables. Execution times are given in clock periods. Models 172 through 174 have 50-nanosecond clock periods. Model 175 has a 25-nanosecond clock period.

TABLE A-1. CENTRAL PROCESSOR INSTRUCTION TIMING (MODELS 172 THROUGH 174)

Instruction Code	Description	Execution Time (Clock Periods)		Timing Notes
		Model 172	Models 173/174	
00xxx	Error exit to MA or program stop	-	-	9
010xK	Return jump to K	36	29	-
011jK	Block copy (Bj) + K words from ECS to CM	-	-	3
012jK	Block copy (Bj) + K words from CM to ECS	-	-	3
013jK	Central exchange jump to (Bj) + K	49	42	-
02ixK	Jump to (Bi) + K	29	22	-
030jK	Branch to K if (Xj) = 0	29	22	1
031jK	Branch to K if (Xj) ≠ 0	29	22	1
032jK	Branch to K if (Xj) positive	29	22	1
033jK	Branch to K if (Xj) negative	29	22	1
034jK	Branch to K if (Xj) in range	29	22	1
035jK	Branch to K if (Xj) out of range	29	22	1
036jK	Branch to K if (Xj) definite	29	22	1
037jK	Branch to K if (Xj) indefinite	29	22	1
04ijK	Branch to K if (Bi) = (Bj)	29	22	1
05ijK	Branch to K if (Bi) ≠ (Bj)	29	22	1
06ijK	Branch to K if (Bi) ≥ (Bj)	29	22	1
07ijK	Branch to K if (Bi) < (Bj)	29	22	1
10ijk	Transmit (Xj) to Xi	10	4	-
11ijk	Logical product of (Xj) and (Xk) to Xi	12	6	-
12ijk	Logical sum of (Xj) and (Xk) to Xi	12	6	-
13ijk	Logical difference of (Xj) and (Xk) to Xi	12	6	-
14ixk	Transmit complement of (Xk) to Xi	10	4	-
15ijk	Logical product of (Xj) and comp of (Xk) to Xi	12	6	-
16ijk	Logical sum of (Xj) and comp of (Xk) to Xi	12	6	-
17ijk	Logical difference of (Xj) and comp of (Xk) to Xi	12	6	-

TABLE A-1. CENTRAL PROCESSOR INSTRUCTION TIMING (MODELS 172 THROUGH 174) (Cont'd)

Instruction Code	Description	Execution Time (Clock Periods)		Timing Notes
		Model 172	Models 173/174	
20ijk	Left shift (Xi) by jk	12	6	-
21ijk	Right shift (Xi) by jk	12	6	-
22ijk	Left shift (Xk) nominally (Bj) places to Xi	12	6	-
23ijk	Right shift (Xk) nominally (Bj) places to Xi	12	6	-
24ijk	Normalize (Xk) to Xi and Bj	13	7	-
25ijk	Round normalize (Xk) to Xi and Bj	13	7	-
26ijk	Unpack (Xk) to Xi and Bj	12	6	-
27ijk	Pack (Xk) and (Bj) to Xi	12	6	-
30ijk	Floating sum of (Xj) and (Xk) to Xi	17	11	-
31ijk	Floating difference of (Xj) and (Xk) to Xi	17	11	-
32ijk	Floating DP sum of (Xj) and (Xk) to Xi	17	11	-
33ijk	Floating DP difference of (Xj) and (Xk) to Xi	17	11	-
34ijk	Round floating sum of (Xj) and (Xk) to Xi	17	11	-
35ijk	Round floating difference of (Xj) and (Xk) to Xi	17	11	-
36ijk	Integer sum of (Xj) and (Xk) to Xi	12	6	-
37ijk	Integer difference of (Xj) and (Xk) to Xi	12	6	-
40ijk	Floating product of (Xj) and (Xk) to Xi	64	58	-
41ijk	Round floating product of (Xj) and (Xk) to Xi	64	58	-
42ijk	Floating DP product of (Xj) and (Xk) to Xi	64	58	-
43ijk	Form mask of jk bits to Xi	12	6	-
44ijk	Floating divide (Xj) by (Xk) to Xi	64	58	-
45ijk	Round floating divide (Xj) by (Xk) to Xi	64	58	-
46000	No operation (pass)	10	3	-
464jK	Move indirect	-	-	4, 6
465	Move direct	-	-	4, 5
466	Compare collated	-	-	4, 8
467	Compare uncollated	-	-	4, 7
47ixk	Population count of (Xk) to Xi	73	67	-
50ijK	Set Ai to (Aj) + K	-	-	2
51ijK	Set Ai to (Bj) + K	-	-	2
52ijK	Set Ai to (Xj) + K	-	-	2
53ijk	Set Ai to (Xj) + (Bk)	-	-	2
54ijk	Set Ai to (Aj) + (Bk)	-	-	2
55ijk	Set Ai to (Aj) - (Bk)	-	-	2
56ijk	Set Ai to (Bj) + (Bk)	-	-	2
57ijk	Set Ai to (Bj) - (Bk)	-	-	2
60ijK	Set Bi to (Aj) + K	11	5	-
61ijK	Set Bi to (Bj) + K	11	5	-
62ijK	Set Bi to (Xj) + K	11	5	-
63ijk	Set Bi to (Xj) + (Bk)	11	5	-
64ijk	Set Bi to (Aj) + (Bk)	11	5	-
65ijk	Set Bi to (Aj) - (Bk)	11	5	-
66ijk	Set Bi to (Bj) + (Bk)	11	5	-
67ijk	Set Bi to (Bj) - (Bk)	11	5	-

TABLE A-1. CENTRAL PROCESSOR INSTRUCTION TIMING (MODELS 172 THROUGH 174) (Cont'd)

Instruction Code	Description	Execution Time (Clock Periods)		Timing Notes
		Model 172	Models 173/174	
70ijK	Set Xi to (Aj) + K	12	6	-
71ijK	Set Xi to (Bj) + K	12	6	-
72ijK	Set Xi to (Xj) + K	12	6	-
73ijk	Set Xi to (Xj) + (Bk)	12	6	-
74ijk	Set Xi to (Aj) + (Bk)	12	6	-
75ijk	Set Xi to (Aj) - (Bk)	12	6	-
76ijk	Set Xi to (Bj) + (Bk)	12	6	-
77ijk	Set Xi to (Bj) - (Bk)	12	6	-

Timing Notes:

1. 5 clock periods if jump condition not present.
2. If i = 0, (model 172, 12 clock periods) (models 173 and 174, 5 clock periods).
   If i = 1 through 5, (model 172, 28 clock periods) (models 173 and 174, 21 clock periods).
   If i = 6 or 7, (model 172, 17 clock periods) (models 173 and 174, 10 clock periods).
3. Refer to ECS timing information in Volume 3 of publication number 60347100.
4. Formulas (given in notes 5 through 8) for instruction execution times give only approximate times. The following assumptions make the formulas useful only as best-case calculations.
   a. No offset in either the source field or the destination field (C1=C2=zero).
   b. No memory conflicts from the rest of the system [peripheral processors (PPs), second central processor (CP), or extended core storage (ECS) subsystem].
   c. No memory refresh conflicts or conflicts within the instruction.
   d. All words compare for instruction 467.
   e. 17 clock periods are required following compare/move instructions to complete next instruction word RNI.

NOTE

Formula term explanations for notes 5 through 8 are:

T =	Time required for instruction execution in nanoseconds
L =	Number of characters in the operation
N =	Word count, calculated as L/10
X =	Number of collate operations which require two memory references
Y =	Number of collate operations which require one memory reference
Z =	Number of collate operations which do not require memory references

5. Execution time for model 172:

   T = 1250 + 500N, for N greater than or equal to 5
   T = 2400 for N = 1

   Execution time for models 173 or 174:

   T = 900 + 500N, for N greater than or equal to 5
   T = 2400 for N = 1

TABLE A-1. CENTRAL PROCESSOR INSTRUCTION TIMING
(MODELS 172 THROUGH 174) (Cont'd)

6. Execution time for model 172, 173, or 174:

   T = 1000 + move direct instruction execution time (refer to note 5)

7. Execution time for model 172:

   T = 1150 + 725N, if N is even
   T = 1375 + 725N, if N is odd

   Execution time for model 173 or 174:

   T = 800 + 725N, if N is even
   T = 1025 + 725N, if N is odd

8. Execution time for model 172:

   T = 1150 + 725N + 1600X + 1350Y + 300Z, if N is even
   T = 1375 + 725N + 1600X + 1350Y + 300Z, if N is odd

   Execution time for model 173 or 174:

   T = 800 + 725N + 1600X + 1350Y + 300Z, if N is even
   T = 1025 + 725N + 1600X + 1350Y + 300Z, if N is odd

9. When used as error exit, 00 instructions take 52 clock periods

TABLE A-2. CENTRAL PROCESSOR INSTRUCTION TIMING (MODEL 175)

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
00xxx	Error exit to MA or program stop	-	-	-
010xK	Return jump to K	-	28	1, 2, 3
011jK	Block copy (Bj) + K words from ECS to CM	-	[(Bj) + K] 4	4, 5, 6, 7, 9
012jK	Block copy (Bj) + K words from CM to ECS	-	[(Bj) + K] 4	4, 5, 6, 7, 9
013jK	Central exchange jump to (Bj) + K (monitor flag set)	-	91	1, 2, 4
013xx	Central exchange jump to MA (monitor flag not set)	-	91	1, 2, 4
02ixK	Jump to (Bi) + K	-	26	1, 2, 3, 8, 18
030jK	Branch to K if (Xj) = 0	-	26	1, 2, 3, 10, 11, 18
031jK	Branch to K if (Xj) $\neq$ 0	-	26	1, 2, 3, 10, 11, 18
032jK	Branch to K if (Xj) positive	-	26	1, 2, 3, 10, 11, 18
033jK	Branch to K if (Xj) negative	-	26	1, 2, 3, 10, 11, 18
034jK	Branch to K if (Xj) in range	-	26	1, 2, 3, 10, 11, 18
035jK	Branch to K if (Xj) out of range	-	26	1, 2, 3, 10, 11, 18
036jK	Branch to K if (Xj) definite	-	26	1, 2, 3, 10, 11, 18
037jK	Branch to K if (Xj) indefinite	-	26	1, 2, 3, 10, 11, 18
04ijK	Branch to K if (Bi) = (Bj)	-	26	1, 2, 3, 10, 11, 18
05ijK	Branch to K if (Bi) $\neq$ (Bj)	-	26	1, 2, 3, 10, 11, 18
06ijK	Branch to K if (Bi) $\geq$ (Bj)	-	26	1, 2, 3, 10, 11, 18
071ijK	Branch to K if (Bi) < (Bj)	-	26	1, 2, 3, 10, 11, 18
10ijx	Transmit (Xj) to Xi	Boolean	2	8, 12, 13
11ijk	Logical product of (Xj) and (Xk) to Xi	Boolean	2	8, 12, 13

TABLE A-2. CENTRAL PROCESSOR INSTRUCTION TIMING (MODEL 175) (Cont'd)

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
12ijk	Logical sum of (Xj) and (Xk) to Xi	Boolean	2	8, 12, 13
13ijk	Logical difference of (Xj) and (Xk) to Xi	Boolean	2	8, 12, 13
14ixk	Transmit complement to (Xk) to Xi	Boolean	2	8, 12, 13
15ijk	Logical product of (Xj) and complement of (Xk) to Xi	Boolean	2	8, 12, 13
16ijk	Logical sum of (Xj) and complement of (Xk) to Xi	Boolean	2	8, 12, 13
17ijk	Logical differerence of (Xj) and complement of (Xk) to Xi	Boolean	2	8, 12, 13
20ijk	Left shift (Xi) by jk	Shift	2	8, 12, 13
21ijk	Right shift (Xi) by jk	Shift	2	8, 12, 13
22ijk	Left shift (Xk) nominally (Bj) places to Xi	Shift	2	8, 12, 13
23ijk	Right shift (Xk) nominally (Bj) places to Xi	Shift	2	8, 12, 13
24ijk	Normalize (Xk) to Xi and Bj	Normalize	3	8, 12, 13
25ijk	Round normalize (Xk) to Xi and Bj	Normalize	3	8, 12, 13
26ijk	Unpack (Xk) to Xi and Bj	Boolean	2	8, 12, 13
27ijk	Pack (Xk) and (Bj) to Xi	Boolean	2	8, 12, 13
30ijk	Floating sum of (Xj) and (Xk) to Xi	Floating add	4	8, 12, 13
31ijk	Floating difference of (Xj) and (Xk) to Xi	Floating add	4	8, 12, 13

TABLE A-2. CENTRAL PROCESSOR INSTRUCTION TIMING (MODEL 175) (Cont'd)

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
32ijk	Floating double-precision sum of (Xj) and (Xk) to Xi	Floating add	4	8, 12, 13
33ijk	Floating double-precision difference of (Xj) and (Xk) to Xi	Floating add	4	8, 12, 13
34ijk	Round floating sum of (Xj) and (Xk) to Xi	Floating add	4	8, 12, 13
35ijk	Round floating difference of (Xj) and (Xk) to Xi	Floating add	4	8, 12, 13
36ijk	Integer sum of (Xj) and (Xk) to Xi	Long add	2	8, 12, 13
37ijk	Integer difference of (Xj) and (Xk) to Xi	Long add	2	8, 12, 13
40ijk	Floating product of (Xj) and (Xk) to Xi	Multiply	5	8, 12, 13, 14
41ijk	Round floating product of (Xj) and (Xk) to Xi	Multiply	5	8, 12, 13, 14
42ijk	Floating double-precision product of (Xj) and (Xk) to Xi	Multiply	5	8, 12, 13, 14
43ijk	Form mask of jk bits to Xi	Shift	2	8, 12, 13
44ijk	Floating divide (Xj) by (Xk) to Xi	Divide	20	8, 12, 13, 15
45ijk	Round floating divide (Xj) by (Xk) to Xi	Divide	20	8, 12, 13, 15
460xx	Pass	-	1	
47ixk	Population count of (Xk) to Xi	Pop. Count	2	8, 12, 13
50ijK	Set Ai to (Aj)+K	Increment	23	2, 3, 8, 16, 17, 18
51ijK	Set Ai to (Bj)+K	Increment	23	2, 3, 8, 16, 17, 18
52ijK	Set Ai to (Xj)+K	Increment	23	2, 3, 8, 16, 17, 18
53ijk	Set Ai to (Xj)+(Bk)	Increment	23	2, 3, 8, 16, 17, 18

TABLE A-2. CENTRAL PROCESSOR INSTRUCTION TIMING (MODEL 175) (Cont'd)

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
54ijk	Set Ai to (Aj)+(Bk)	Increment	23	2, 3, 8, 16, 17, 18
55ijk	Set Ai to (Aj) - (Bk)	Increment	23	2, 3, 8, 16, 17, 18
56ijk	Set Ai to (Bj)+(Bk)	Increment	23	2, 3, 8, 16, 17, 18
57ijk	Set Ai to (Bj) - (Bk)	Increment	23	2, 3, 8, 16, 17, 18
60ijK	Set Bi to (Aj)+K	Increment	2	8, 12, 13
61ijK	Set Bi to (Bj)+K	Increment	2	8, 12, 13
62ijK	Set Bi to (Xj)+K	Increment	2	8, 12, 13
63ijk	Set Bi to (Xj)+(Bk)	Increment	2	8, 12, 13
64ijk	Set Bi to (Aj)+(Bk)	Increment	2	8, 12, 13
65ijk	Set Bi to (Aj) - (Bk)	Increment	2	8, 12, 13
66ijk	Set Bi to (Bj)+ (Bk)	Increment	2	8, 12, 13
67ijk	Set Bi to (Bj) - (Bk)	Increment	2	8, 12, 13
70ijK	Set Xi to (Aj)+K	Increment	2	8, 12, 13
71ijK	Set Xi to (Bj)+K	Increment	2	8, 12, 13
72ijK	Set Xi to (Xj)+K	Increment	2	8, 12, 13
73ijk	Set Xi to (Xj)+(Bk)	Increment	2	8, 12, 13
74ijk	Set Xi to (Aj)+(Bk)	Increment	2	8, 12, 13
75ijk	Set Xi to (Aj) - (Bk)	Increment	2	8, 12, 13
76ijk	Set Xi to (Bj)+(Bk)	Increment	2	8, 12, 13
77ijk	Set Xi to (Bj) - (Bk)	Increment	2	8, 12, 13

TABLE A-2. CENTRAL PROCESSOR INSTRUCTION TIMING (MODEL 175) (Cont'd)

Timing Notes:

1. All previous instruction fetches are completed.
2. No CM conflicts or SAS backup caused by CM conflicts exist.
3. No PPS request occurs.
4. All operating registers are free.
5. ECS is not busy.
6. All ECS banks have completed previously initiated read/write cycles.
7. Time does not include start-up time.
8. The requested operating register(s) is free.
9. Time assumes no ECS record gaps.
10. If the address is in the IAS, the execution time is 3 clock periods.
11. If the branch conditions are not met, the execution time is 2 clock periods.
12. The requested destination register(s) input data path is free during the required clock period.
13. After the instruction has issued to the functional unit, no further delay is possible.
14. The multiply unit is free.
15. The divide unit is free.
16. If i=0, execution time is 2 clock periods, and no storage reference is required. If i=1 through 5, execution time is 23 clock periods, and a storage reference is required. If i=6 or 7, execution time is 2 clock periods, and a storage reference continues after instruction execution.
17. After the instruction has issued to the increment unit, no further delays are possible in the delivery of data to the Ai register. However, CM conflicts may delay the resulting storage reference.
18. If memory enable is present when the address is gated into SAS, one additional clock period is required.

PERIPHERAL PROCESSOR INSTRUCTION TIMING

Execution times for the peripheral processor (PP) instructions are given in Table A-3. The times listed in the execution time column assume that no conflicts occur. The timing notes column is keyed to the notes listed at the end of the table. Execution times are given in minor cycles (100 nanoseconds in 1X mode, and 50 nanoseconds in 2X mode).

TABLE A-3. PERIPHERAL PROCESSOR INSTRUCTION TIMING

Instruction Code	Description	Execution Time (Minor Cycles)	Timing Notes
0000	Pass	10	
01dm	Long jump to m + (d)	-	1
02dm	Return jump to m + (d)	-	2
03d	Unconditional jump d	10	
04d	Zero jump d	10	
05d	Nonzero jump d	10	
06d	Plus jump d	10	
07d	Minus jump d	10	
10d	Shift d	10	
11d	Logical difference d	10	
12d	Logical product d	10	
13d	Selective clear d	10	
14d	Load d	10	
15d	Load complement d	10	
16d	Add d	10	
17d	Subtract d	10	
20dm	Load dm	20	
21dm	Add dm	20	
22dm	Logical product dm	20	
23dm	Logical difference dm	20	
2400	Pass	10	
2500	Pass	10	
260x	Exchange jump	-	3, 9
261x	Monitor exchange jump	-	3, 9
262x	Monitor exchange jump to MA	-	3, 9
27x	Read program address	10	
30d	Load (d)	20	
31d	Add (d)	20	
32d	Subtract (d)	20	
33d	Logical difference (d)	20	
34d	Store d	20	
35d	Replace add (d)	30	
36d	Replace add one (d)	30	
37d	Replace subtract one (d)	30	
40d	Load ((d))	30	
41d	Add ((d))	30	
42d	Subtract ((d))	30	
43d	Logical difference ((d))	30	
44d	Store ((d))	30	
45d	Replace add ((d))	40	
46d	Replace add one ((d))	40	
47d	Replace subtract one ((d))	40	

TABLE A-3. PERIPHERAL PROCESSOR INSTRUCTION TIMING (Cont'd)

Instruction Code	Description	Execution Time (Minor Cycles)	Timing Notes
50dm	Load (m + (d))	-	2
51dm	Add (m + (d))	-	2
52dm	Subtract (m + (d))	-	2
53dm	Logical difference (m + (d))	-	2
54dm	Store (m + (d))	-	2
55dm	Replace add (m + (d))	-	4
56dm	Replace add one (m + (d))	-	4
57dm	Replace subtract one (m + (d))	-	4
60d	Central read from (A) to d	-	5, 8
61dm	Central read (d) words to (A) from m	-	6, 8
62d	Central write to (A) from d	-	5, 8
63dm	Central write (d) words to (A) from m	-	6, 8
64dm	Jump to m if channel d active	20	
65dm	Jump to m if channel d inactive	20	
66dm	Jump to m if channel d full	20	
67dm	Jump to m if channel d empty	20	
70d	Input to A from channel d	20	
71dm	Input (A) words to m from channel d	-	7, 10
72d	Output from A on channel d	20	
73dm	Output (A) words from m on channel d	-	7, 10
74d	Activate channel d	20	
75d	Disconnect channel d	20	
76d	Function (A) on channel d	20	
77dm	Function m on channel d	20	

Timing Notes:

1. 30 cycles; if d = 0, 20 cycles.
2. 40 cycles; if d = 0, 30 cycles.
3. Executes in 10 cycles if no CMC conflict occurs.
4. 50 cycles; if d = 0, 40 cycles.
5. Minimum of 60 cycles.
6. 60 cycles plus 50 cycles per word.
7. 50 cycles plus 10 cycles per word.
8. Conflicts in CM cause indeterminate delays.
9. Following an exchange jump instruction, the CP must complete the exchange jump before further PPM references or exchange jump instructions can be executed.
10. Instructions for input/output (I/O) and for PPM references can transfer a word every 10 cycles although the peripheral equipment seldom permits this rate for I/O operations.

GLOSSARY

CEJ	Central exchange jump
CIW	Current instruction word
CM	Central memory
CMC	Central memory control
CP	Central processor
CPU	Central processing unit
CSU	Central storage unit
ECS	Extended core storage
EM	Exit mode
FLC	Field length for CM
FLE	Field length for ECS
IAS	Instruction address stack
IFA	Instruction fetch address
I/O	Input/output
IWS	Instruction word stack
MA	Monitor address
MEJ	Monitor exchange jump
MF	Monitor flag
MOS	Metal oxide semiconductor
P	Program address
PE	Parity error
PP	Peripheral processor
PPM	Peripheral processor memory
PPS	Peripheral processor subsystem
RAC	Reference address for CM
RAE	Reference address for ECS
RNI	Read next instruction
RVM	Reference voltage margin
SAS	Storage address stack
SECDED	Single-error correction double-error detection
SRO	Storage read out
SWS	Storage word stack

NOTE

Instruction designators are defined in Tables 4-1 and 4-3.

INDEX

COMMENT SHEET

MANUAL TITLE CDC CYBER 170 Computer System

Hardware Reference Manual

PUBLICATION NO. 60420000 REVISION C

FROM: NAME:

BUSINESS ADDRESS:

CUT ALONG LINE

PRINTED IN U.S.A.

AA3419 REV. 7/75

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY
CONTROL DATA CORPORATION
Publications and Graphics Division
ARH219
4201 North Lexington Avenue
Saint Paul, Minnesota 55112

CUT ALONG LINE

FOLD

FOLD

CORPORATE HEADQUARTERS, P.O. BOX 0, MINNEAPOLIS, MINNESOTA 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

LITHO IN U.S.A.

CONTROL DATA CORPORATION