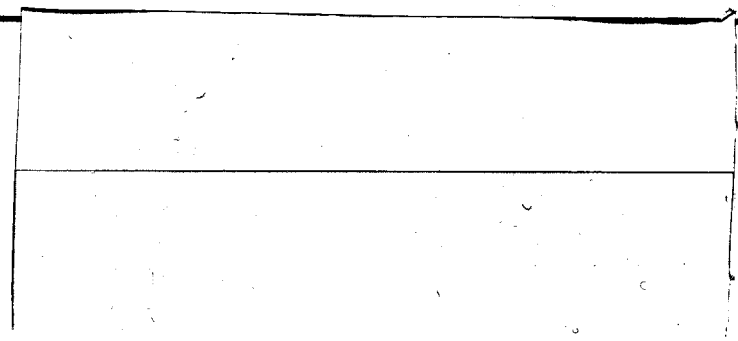
60420000

CONTROL DATA CORPORATION



HARDWARE REFERENCE MANUAL

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	Manua -4, 5-5, and A-1 through
(7-7-75)	A-10
C	Manua ges 2-57 and 5-40 are revised.
(9-1-75)	
D	This r rder 36696.
(11-1-76)	
E	Manua ual obsoletes all previous
(11-1-76)	edition 9983600, and includes
	the inf
F	Manual revised; includes Engineering Change Order 37879 which updates models 172 through 175
(4-1-77)	information and adds preliminary information for models 171 and 176. Because of extensive
	changes to this manual, chart tape and dots are not used, and all pages reflect the latest revision
	level. This manual obsoletes all previous editions.
G	Manual revised; includes Engineering Change Order 37761. Pages 3-3 and 3-4 are revised.
(8-24-77)	
H	Manual revised; includes Engineering Change Orders 37767 and 37971. Adds information for
(8-24-77)	models 171C through 175C. This edition obsoletes all previous editions.
Publication No. 60420000	

REVISION LETTERS I, O, Q AND X ARE NOT USED

Printed in the United States of America

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.

LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV
Front Cover	-
Inside Cover	-
Title Page	-
ii	H
iii/iv	H
v	H
vi	H
vii	H
viii	H
ix	H
x	H
xi	H
1-1/1-2	F
Divider	-
1-3	H
1-4	H
1-5	H
1-6	H
1-7	H
1-8	H
1-9/1-10	H
Divider	-
1-11	H
1-12	H
1-13	H
1-14	H
1-15	H
1-16	H
1-17	H
1-18	H
1-19/1-20	H
Divider	-
1-21	H
1-22	H
1-23	H
1-24	H
2-1/2-2	F
Divider	-
2-3	H
2-4	H
2-5	F
2-6	F
2-7/2-8	F
Divider	-
2-9	H
2-10	H
2-11	F
2-12	F
2-13	F
2-14	H
2-15	F
2-16	F
Divider	-
2-17	H
2-18	H
2-19	F
2-20	H
2-21/2-22	H
Divider	-
2-23	H
2-24	H
2-25/2-26	H
Divider	-
2-27	H
2-28	H
Divider	-
2-29	F
2-30	H
Divider	-
2-31	F
2-32	F
2-33	H
2-34	F
Divider	-
2-35/2-36	F
Divider	-
2-37	F
2-38	F
Divider	-
2-39/2-40	F
Divider	-
2-41	F
2-42	F
2-43	F
2-44	F
Divider	-
2-45	H
2-46	H
2-47	F
3-1/3-2	F
Divider	-
3-3	G
3-4	G
3-5	H
3-6	F
3-7	H
3-8	H
3-9	H
3-10	H
3-11	H
3-12	H
3-13	H
3-14	H
3-15	H
3-16	H
Divider	-
3-17	H
3-18	F
Divider	-
3-19	H
3-20	H
3-21	H
3-22	H
3-23	F
3-24	F
3-25	H
3-26	F
Divider	-
3-27	H
3-28	H
4-1/4-2	F
Divider	-
4-3	F
4-4	F
4-5	F
4-6	F
4-7	F
4-8	H
4-9	F
4-10	F
4-11	F
4-12	F
4-13	F
4-14	F
4-15	F
4-16	F
4-17	F
4-18	F
4-19	F
4-20	F
4-21	F
4-22	F
4-23	H
4-24	F
4-25	F
4-26	F
4-27	F
4-28	F
4-29	F
4-30	F
4-31	F
4-32	F
4-33	H
4-34	F
4-35	F
4-36	F
4-37	F
4-38	F
4-39	F
4-40	F
Divider	-
4-41	H
4-42	F
4-43	F
4-44	F
4-45	F
4-46	F
4-47	F
4-48	F
4-49	F
4-50	F
4-51	F
4-52	F
4-53	F
4-54	F
4-55	F
4-56	F
4-57	F
4-58	F
4-59/4-60	F
Divider	-
4-61	F
4-62	F
4-63	F
4-64	F
4-65	F
4-66	H
4-67	F
4-68	F
4-69	F
4-70	F
4-71	F
4-72	H
5-1/5-2	H
Divider	-
5-3	F
5-4	F
5-5	F
5-6	F
5-7	F
5-8	F
5-9	F
5-10	F
5-11	F
5-12	F
5-13	F
5-14	F
5-15	F
5-16	F
5-17	F
5-18	F
5-19	F
5-20	F
5-21/5-22	F
Divider	-
5-23	F
5-24	F
Divider	-
5-25	H
5-26	H
5-27	H
5-28	F
5-29	F
5-30	H
5-31/5-32	F
Divider	-
5-33	F
5-34	F
5-35/5-36	F
Divider	-
5-37	F
5-38	F
Divider	-
5-39	F
5-40	F
5-41	F
5-42	F
5-43	H
5-44	H
5-45	H
5-46	H
5-47	H
5-48	H
Divider	-
5-49	H
5-50	F
5-51	F
5-52	F
5-53	F
5-54	F
5-55	H
5-56	F
5-57	H
5-58	H
5-59	F
5-60	H
5-61	F
5-62	F
5-63	H
5-64	H
5-65	F
5-66	H
5-67/5-68	F
Divider	-
5-69	F
5-70	F
5-71	F
5-72	H
5-73	H
5-74	H
5-75	F
5-76	H
5-77	F
5-78	F
5-79	H
5-80	H
5-81	H
5-82	F
5-83	F
A-1	F
B-1	F
B-2	F
Index-1	H
Index-2	H
Comment Sheet	H
Back Cover	-

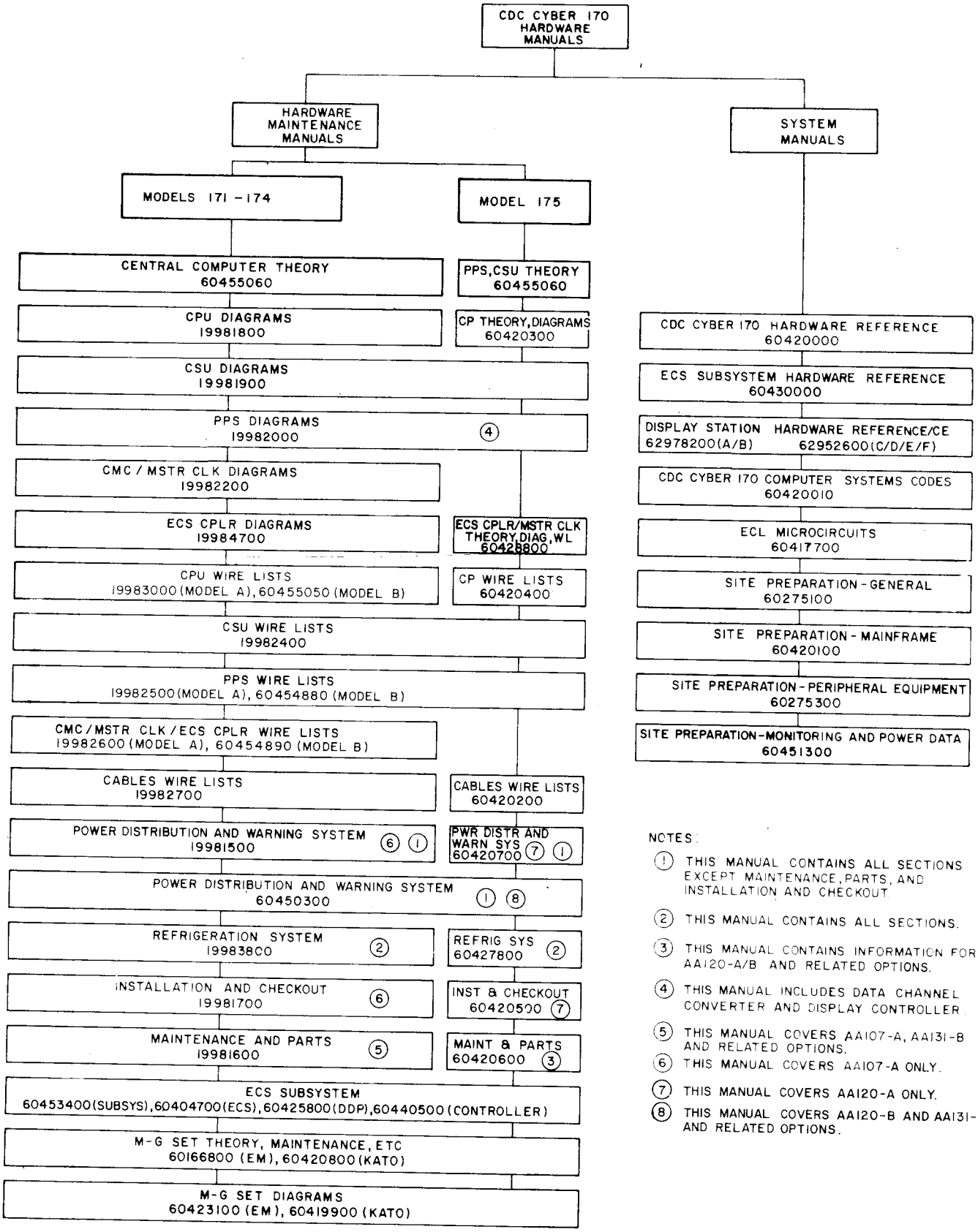
PREFACE

This manual contains hardware reference information for the CONTROL DATA® CYBER 170 Computer Systems, models 171 through 176. These model numbers are used throughout the manual to identify the reference information. In some parts of the manual, designators A, B, and C are used to identify information which is characteristic of systems produced after different production cut-in times. For instance, designator A relates to the original production models, B to later models, and C to the most recent models.

The manual describes the functional, operational, and programming characteristics of the computer systems hardware. Additional system hardware information is available for models 171 through 175 in the publications listed in the system publication index on the following page. Model 176 publications will be added to the index as they become available.

This manual is for use by customer, marketing, training, programming, and Engineering Services personnel who operate, program, and maintain the computer systems.

Publication ordering information and latest revision levels are available from the Literature Distribution Services catalog, publication number 90310500.



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 AA120-A/B AND RELATED OPTIONS.

(4) THIS MANUAL INCLUDES DATA CHANNEL CONVERTER AND DISPLAY CONTROLLER.

(5) THIS MANUAL COVERS AA107-A, AA131-B AND RELATED OPTIONS.

(6) THIS MANUAL COVERS AA107-A ONLY.

(7) THIS MANUAL COVERS AA120-A ONLY.

(8) THIS MANUAL COVERS AA120-B AND AA131-B AND RELATED OPTIONS.

SYSTEM PUBLICATION INDEX

677

CONTENTS

APPENDIXES

INDEX

FIGURES

TABLES

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

INTRODUCTION

The CDC CYBER 170 systems (figure 1-1) include models 171 through 176. These are general purpose digital computer systems that provide varying degrees of processing power, data storage, and input/output (I/O) capabilities.

Depending upon options and design differences, the systems include one or more of the following components.

- Central processor (CP)
- Central memory control (CMC) in models 171 through 175 and memory control in model 176
- Central memory (CM), includes one or two central storage units (CSUs) in models 171 through 175 and small semiconductor memory (SSM) in model 176
- Large core memory extension (LCME) in model 176
- Extended core storage (ECS), optional, in models 171 through 175
- Peripheral processor subsystem (PPS), includes 10 peripheral processors (PPs)
- Peripheral processor units (PPUs) in model 176
- Data channel converter (DCC)
- Display station
- Condensing unit(s)
- Power distribution unit (PDU)

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).

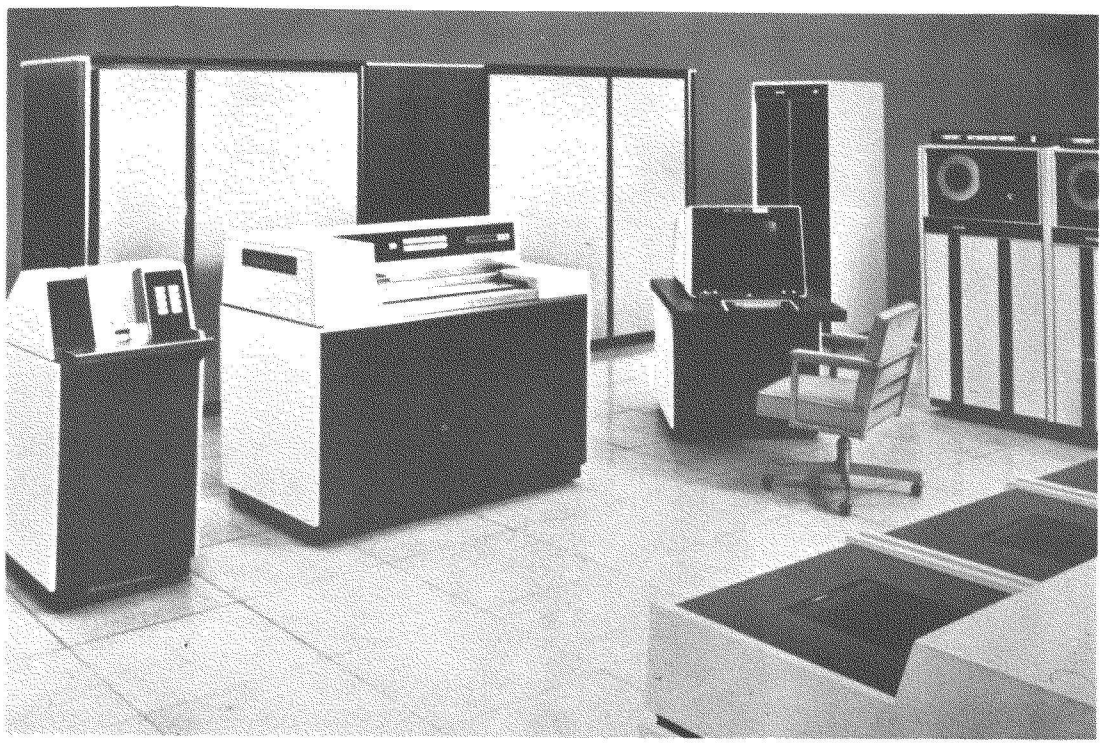


Figure 1-1. Typical CDC CYBER 170 System

PHYSICAL CHARACTERISTICS

PHYSICAL CHARACTERISTICS

Many components of the system models are functionally the same or similar. For these components, the manual provides a common description. Components with different functions have individual descriptions that are identified by the system model number (table 1-1).

TABLE 1-1. CDC CYBER 170 SYSTEM COMPONENTS

Components	Models					
	171	172	173	174	175	176
Mainframe:						
Central processor-0	x	x	x	x	x	x
Central processor-1	*	*	-	x	-	-
Compare/move unit for CP-0	*	x	x	x	-	-
Compare/move unit for CP-1 (optional compare/move unit must be installed for model 171)	*	x	-	x	-	-
Central memory control	x	x	x	x	x	-
Memory control	-	-	-	-	-	x
Central memory, eight banks in CSU-0	x	x	x	x	x	-
Central memory, eight banks in CSU-1	*	*	*	*	x	-
Central memory, 16 banks	-	-	-	-	-	x
Peripheral processor subsystem-0	x	x	x	x	x	x
Peripheral processor subsystem-1	*	*	*	*	*	*
Peripheral processor units	-	-	-	-	-	x
I/O multiplexer	-	-	-	-	-	x
Logic scanner	-	-	-	-	-	x
One data channel converter for PPS-0	*	-	-	-	-	x
Two data channel converters for PPS-0	*	x	x	x	x	-
Two data channel converters for PPS-1	*	*	*	*	*	*
Display controller	x	x	x	x	x	x
Extended core storage coupler	*	*	*	*	*	-
One 3-ton internally mounted condensing unit	x	x	x	-	-	-
Two 3-ton internally mounted condensing units	*	*	*	x	-	-
Large core memory extension	-	-	-	-	-	x
Extended core storage subsystem	*	*	*	*	*	-
One 10-ton externally mounted condensing unit	-	-	-	-	x	*
Two 10-ton externally mounted condensing units	-	-	-	-	-	x
Power distribution unit	-	-	-	-	-	x
Display station (first)	x	x	x	x	x	x
Display station (second)	*	*	*	*	*	*

x Standard
- Not available
* Optional

The following model configurations describe the physical arrangements of cabinets, bays, and chassis in basic and maximally configured systems. Additional physical characteristics of the computer systems are on data sheets in the CDC CYBER 170 Section 2 Site Preparation Manual, listed in the preface. The data sheets include separate descriptions of the mainframe models, associated condensing units, and display station. The sheets also include weight, power consumption, and certain code requirements.

MODEL 171 CONFIGURATION

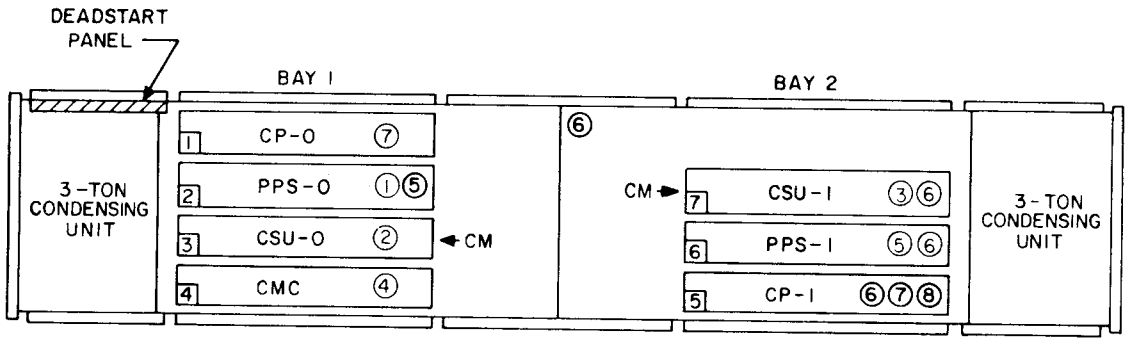
The model 171 basic configuration (figure 1-2) includes a display station and mainframe bay 1, which contains a condensing unit and four chassis for logic and memory modules. The maximum configuration includes a second mainframe bay 2. This bay contains a second condensing unit and up to three chassis for additional logic and memory modules. Installation of the optional ECS requires the addition of a stand-alone cabinet for a controller and from one to four cabinets for the ECS, depending upon the options.

MODEL 172 CONFIGURATION

The model 172 basic configuration (figure 1-3) includes a display station and mainframe bay 1, which contains a condensing unit and four chassis for logic and memory modules. This configuration is the maximum configuration of the original model 172 system. Development of additional equipment options makes later model 172 systems available in a maximum configuration that includes a mainframe bay 2. This bay contains a second condensing unit and up to three chassis for additional logic and memory modules. Installation of the optional ECS requires the addition of a stand-alone cabinet for a controller and from one to four cabinets for the ECS, depending upon the options.

MODEL 173 CONFIGURATION

The model 173 basic configuration (figure 1-4) includes a display station and mainframe bay 1, which contains a condensing unit and four chassis for logic and memory modules. The maximum configuration includes a mainframe bay 2. This bay contains a second condensing unit and up to two chassis for additional logic and memory modules. Installation of the optional ECS requires the addition of a stand-alone cabinet for a controller and from one to four cabinets for the ECS, depending upon the options.



NOTES:

① CHANNEL 2 ALSO CONTAINS A DISPLAY STATION CONTROLLER.

② CSU-0 IS EXPANDABLE FROM 65,536 TO 98,304 TO 131,072 WORDS.

③ CSU-1 EXPANDS MEMORY TO 196,608 TO 262,144 WORDS

④ WHEN ECS IS INSTALLED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

⑤ CHASSIS 2 AND 6 MAY EACH CONTAIN TWO OPTIONAL CHANNEL CONVERTERS.

⑥ BAY 2 AND CHASSIS 5, 6, AND 7 ARE OPTIONAL.

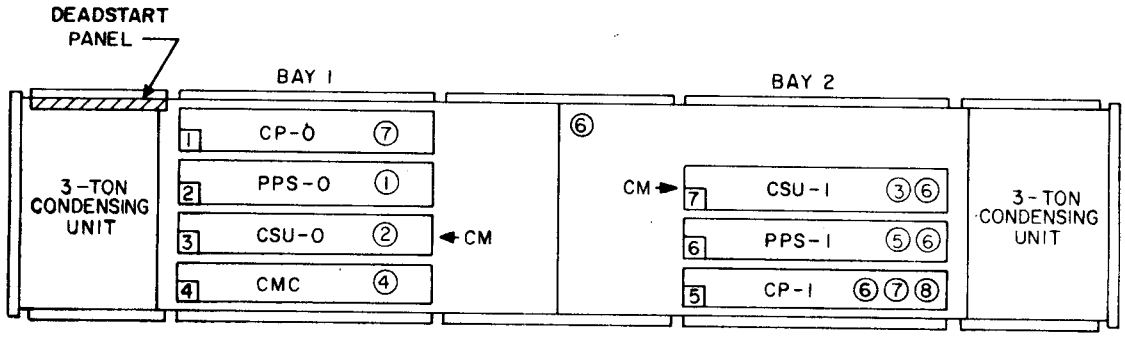
⑦ CHASSIS 1 AND 5 MAY EACH CONTAIN AN OPTIONAL COMPARE/MOVE UNIT.

⑧ IF CHASSIS 5 IS INSTALLED AND CHASSIS 7 IS NOT INSTALLED, CHASSIS 5 MOUNTS IN THE CHASSIS 7 LOCATION.

DISPLAY STATION

6AA208

Figure 1-2. Model 171 Maximum Chassis Configuration (Top Cutaway View)



NOTES:

① CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY STATION CONTROLLER.

② CSU-0 IS EXPANDABLE FROM 65,536 TO 98,304 TO 131,072 WORDS.

③ CSU-1 EXPANDS MEMORY TO 196,608 TO 262,144 WORDS.

④ WHEN ECS IS INSTALLED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

⑤ CHASSIS 6 MAY CONTAIN TWO OPTIONAL DATA CHANNEL CONVERTERS.

⑥ BAY 2 AND CHASSIS 5, 6, AND 7 ARE OPTIONAL.

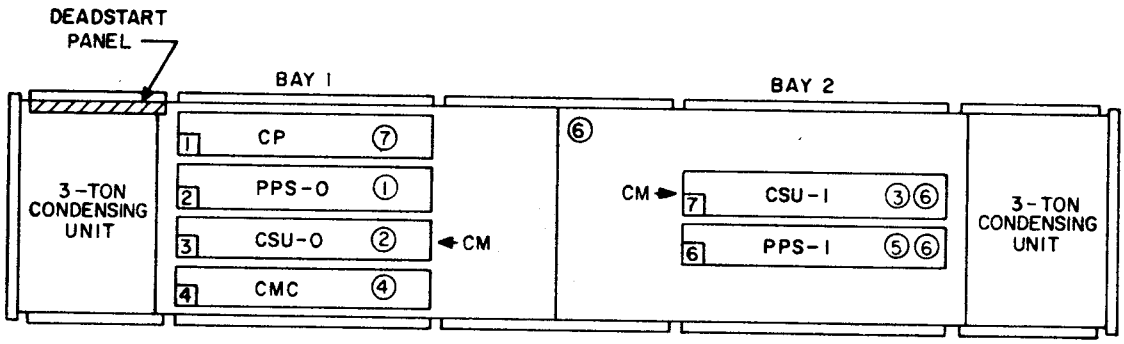
⑦ CHASSIS 1 AND 5 EACH CONTAIN A COMPARE/MOVE UNIT.

⑧ IF CHASSIS 5 IS INSTALLED AND CHASSIS 7 IS NOT INSTALLED, CHASSIS 5 MOUNTS IN THE CHASSIS 7 LOCATION.

DISPLAY STATION

6AA3D

Figure 1-3. Model 172 Maximum Chassis Configuration (Top Cutaway View)



NOTES:

① CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY STATION CONTROLLER.

② CSU-0 IS EXPANDABLE FROM 65,536 TO 98,304 TO 131,072 WORDS.

③ CSU-1 EXPANDS MEMORY TO 196,608 TO 262,144 WORDS.

④ WHEN ECS IS INSTALLED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

⑤ CHASSIS 6 MAY CONTAIN TWO OPTIONAL DATA CHANNEL CONVERTERS.

⑥ BAY 2 AND CHASSIS 6 AND 7 ARE OPTIONAL.

⑦ CHASSIS 1 ALSO CONTAINS A COMPARE/MOVE UNIT.

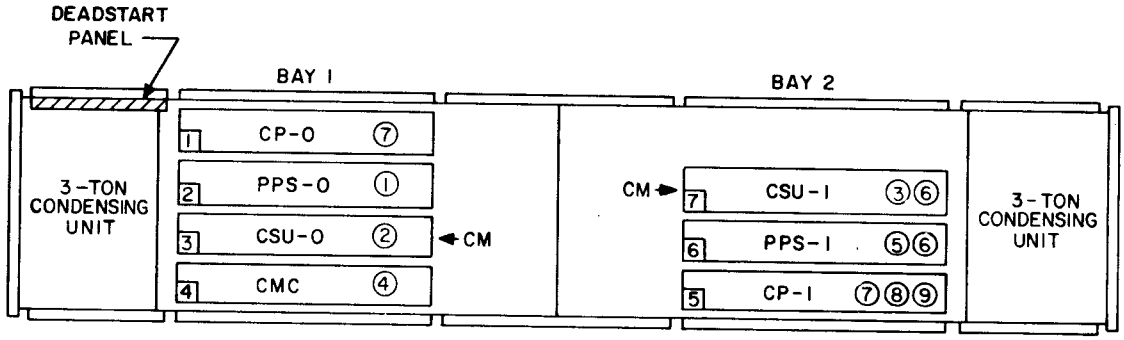
DISPLAY STATION

6AA4D

Figure 1-4. Model 173 Maximum Chassis Configuration (Top Cutaway View)

MODEL 174 CONFIGURATION

The model 174 basic configuration (figure 1-5) includes a display station and mainframe bays 1 and 2, which contain two condensing units, four chassis in bay 1, and one chassis in bay 2. The maximum configuration includes up to two additional chassis in bay 2 for logic and memory modules. Installation of the optional ECS requires the addition of a stand-alone cabinet for a controller and from one to four cabinets for the ECS, depending upon the options.



NOTES:

① CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY STATION CONTROLLER.

② CSU-0 IS EXPANDABLE FROM 65,536 TO 98,304 TO 131,072 WORDS.

③ CSU-1 EXPANDS MEMORY TO 196,608 TO 262,144 WORDS.

④ WHEN ECS IS INSTALLED, CHASSIS 4 ALSO CONTAINS THE ECS COUPLER.

⑤ CHASSIS 6 MAY CONTAIN TWO OPTIONAL DATA CHANNEL CONVERTERS.

⑥ CHASSIS 6 AND 7 ARE OPTIONAL.

⑦ CHASSIS 1 AND 5 EACH CONTAIN A COMPARE/MOVE UNIT.

⑧ CP-1 MAY MOUNT IN CHASSIS 7 LOCATION, IF CSU-1 AND PPS-1 CHASSIS ARE NOT INSTALLED.

⑨ IF CHASSIS 5 IS INSTALLED AND CHASSIS 7 IS NOT INSTALLED, CHASSIS 5 MOUNTS IN THE CHASSIS 7 LOCATION.

DISPLAY STATION

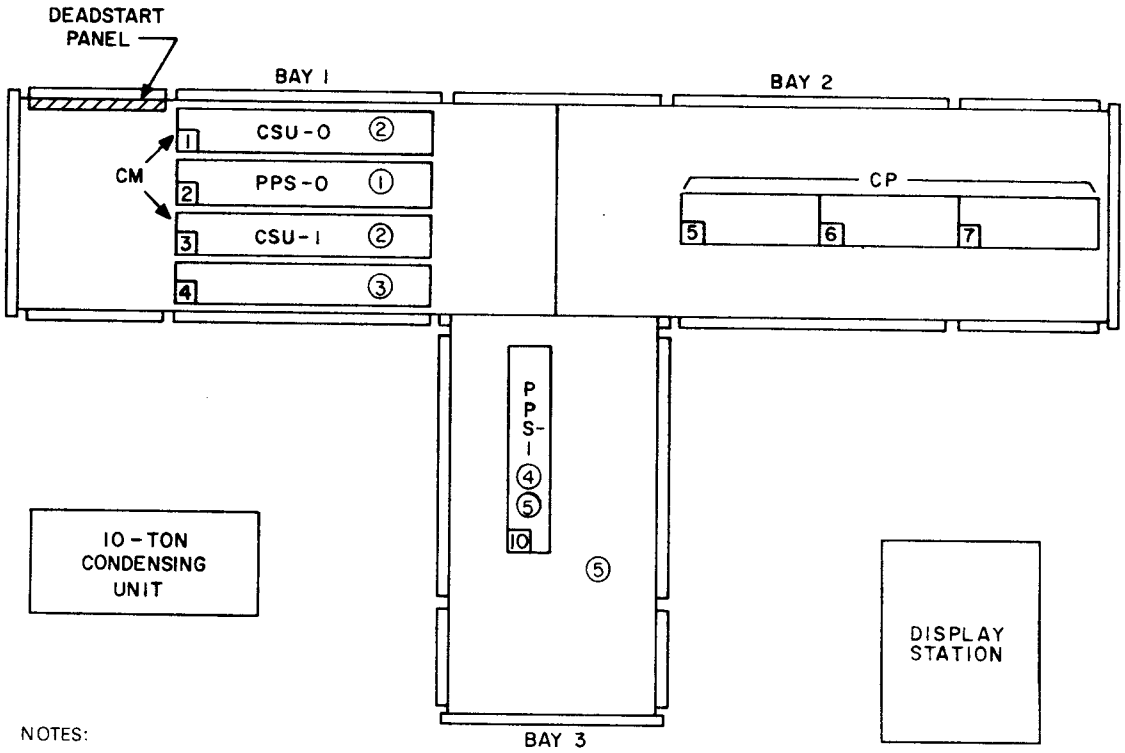
6AA5E

Figure 1-5. Model 174 Maximum Chassis Configuration (Top Cutaway View)

MODEL 175 CONFIGURATION

The models 175 A and B basic configuration (figure 1-6) includes a display station, a stand-alone condensing unit, and mainframe bays 1 and 2, which contain four chassis in bay 1 and three chassis in bay 2.

The maximum configuration includes one additional chassis in bay 3. Installation of the optional ECS requires the addition of a stand-alone cabinet for a controller and from one to four cabinets for the ECS, depending upon the options.



NOTES:

① CHASSIS 2 ALSO CONTAINS TWO DATA CHANNEL CONVERTERS AND A DISPLAY STATION CONTROLLER.

② CSU CHASSIS ARE EXPANDABLE FROM 65,536 TO 98,304 TO 131,072 TO 196,608 TO 262,144 WORDS.

③ WHEN ECS IS INSTALLED, CHASSIS 4 CONTAINS THE ECS COUPLER. 6AA6E

④ CHASSIS 10 MAY CONTAIN TWO OPTIONAL DATA CHANNEL CONVERTERS.

⑤ BAY 3 AND CHASSIS 10 ARE OPTIONAL.

Figure 1-6. Models 175A and B Maximum Chassis Configuration (Top Cutaway View)

The model 175C basic configuration (figure 1-7) includes a display station, a stand-alone condensing unit, and mainframe bays 1 and 2, which contain four chassis in bay 1 and three chassis in bay 2. The maximum configuration has the same bays and chassis except for options added to the chassis. Installation of the optional ECS requires the addition of a stand-alone cabinet for a controller and from one to four cabinets for the ECS, depending upon the options.

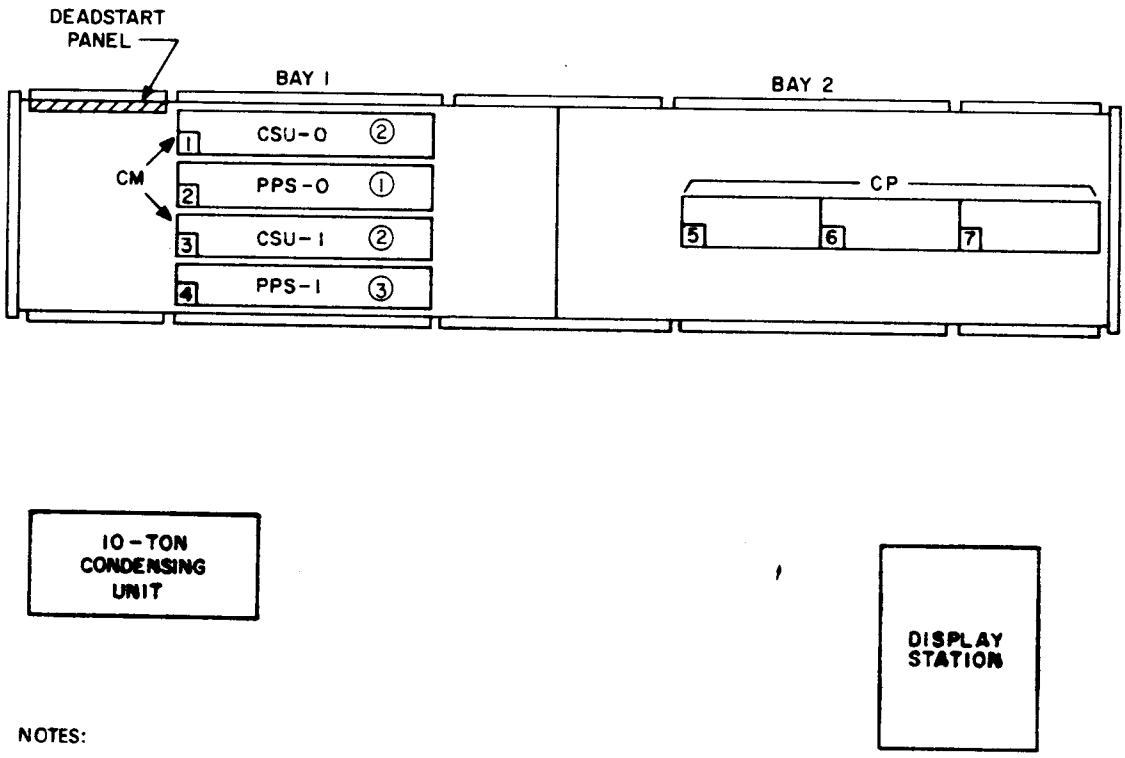


Figure 1-7. Model 175C Maximum Chassis Configuration (Top Cutaway View)

MODEL 176 CONFIGURATION

The model 176 basic configuration (figure 1-8) includes a display station, two condensing units, a stand-alone cabinet with one chassis, and nine mainframe chassis. The maximum configuration includes one additional condensing unit, one additional chassis in the stand-alone cabinet, and five additional mainframe chassis.

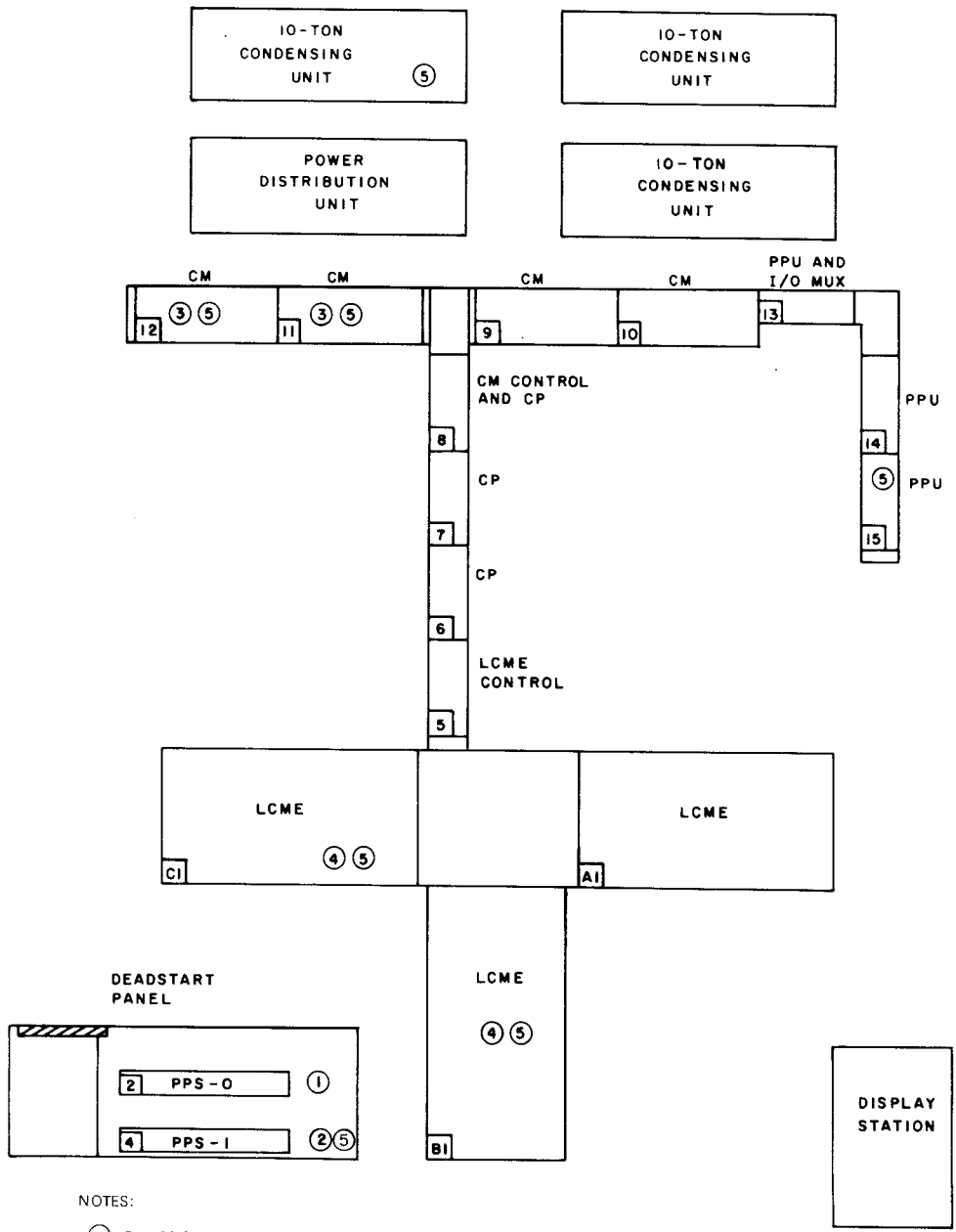


Figure 1-8. Model 176 Maximum Chassis Configuration (Top Cutaway View)

FUNCTIONAL CHARACTERISTICS

FUNCTIONAL CHARACTERISTICS

Tables 1-2 through 1-6 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					
	171	172	173	174	175	176
60-bit internal word	x	x	x	x	x	x
Computation in fixed- and floating-point arithmetic	x	x	x	x	x	x
Eight 60-bit operand X registers	x	x	x	x	x	x
Eight 18-bit address A registers	x	x	x	x	x	x
Eight 18-bit index B registers	x	x	x	x	x	x
Character manipulation by compare/move instructions	*	x	x	x	-	-
Synchronous internal logic with a 50-nanosecond clock period	x	x	x	x	-	-
Synchronous internal logic with a 27.5-nanosecond clock period	-	-	-	-	-	x
Large and small arithmetic sections	x	x	x	x	-	-
Synchronous internal logic with a 25-nanosecond clock period	-	-	-	-	x	-
12-word instruction word stack	-	-	-	-	x	x
Nine functional units	-	-	-	-	x	x

x Standard
- Not available
* Optional

TABLE 1-3. CENTRAL MEMORY FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Model					
	171	172	173	174	175	176
400-nanosecond cycle time	x	x	x	x	x	-
165-nanosecond cycle time for write, 82.5-nanosecond cycle time for read	-	-	-	-	-	x
Maximum transfer rate of one word each 50 nanoseconds	x	x	x	x	x	-
Maximum transfer rate of one word each 27.5 nanoseconds	-	-	-	-	-	x
Semiconductor memory of 65,536 words (60-bit words plus eight error detection/correction bits per word); expandable to 98.304; 131,072; 196,608; and 262,144 words	x	x	x	x	x	-
Semiconductor memory of 131,072 words (60-bit words plus eight error detection/correction bits per word); expandable to 196,608 and 262,144 words	-	-	-	-	-	x
Organized into eight independent banks per CSU	x	x	x	x	-	-
Organized into 16 independent banks	-	-	-	-	x	x

x Standard
- Not available
* Optional

TABLE 1-4. PERIPHERAL PROCESSOR SUBSYSTEM FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Model					
	171	172	173	174	175	176
12-bit internal word	x	x	x	x	x	x
Binary computation in fixed-point arithmetic	x	x	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	x	x
10 PPs time-share access to CM	x	x	x	x	x	x
Each PP has an internal semiconductor memory of 4096 words (12-bit words plus one parity bit per word, odd parity)	x	x	x	x	x	x
12 I/O channels, each accessible by any of the PPs	x	x	x	x	x	x
Status and control register	x	x	x	x	x	x
Real-time clock	x	x	x	x	x	x
Each I/O channel carries 12-bit words plus one parity bit per word (odd parity)	x	x	x	x	x	x
Expandable from 10 to 20 PPs in increments of 4, 3, and 3 and from 12 to 24 I/O channels	*	*	*	*	*	*

x Standard
- Not available
* Optional

TABLE 1-5. PERIPHERAL PROCESSOR UNIT FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Model					
	171	172	173	174	175	176
12-bit internal word	-	-	-	-	-	x
Binary computation in fixed-point arithmetic	-	-	-	-	-	x
27.5-nanosecond clock synchronous with CP	-	-	-	-	-	x
Each PPU has an internal coincident current memory of 4096 words (12-bit words plus one parity bit per word, odd parity)	-	-	-	-	-	x
Eight bidirectional I/O channels dedicated to each PPU	-	-	-	-	-	x

x Standard
- Not available

TABLE 1-6. DATA AND ADDRESS CHECKING FUNCTIONAL CHARACTERISTICS

Functional Characteristics	Model					
	171	172	173	174	175	176
Parity check data between CP-0 and CMC	x	x	x	x	-	-
Parity check data between CP-1 and CMC	x	x	-	x	-	-
Parity check data between PPS-0 and CMC	x	x	x	x	x	-
Parity check data between PPS-1 and CMC	x	x	x	x	x	-
Parity check data between ECS and CMC (only if a parity-enhanced controller is installed)	x	x	x	x	x	-
Single-error correction double-error detection (SECDED) between CM and CMC	x	x	x	x	x	-
SECDED between CM and control	-	-	-	-	-	x
Parity check address from CP-0 to CMC	x	x	x	x	-	-
Parity check address from CP-1 to CMC	x	x	-	x	-	-
Parity check address from PPS-0 to CMC	x	x	x	x	x	-
Parity check address from PPS-1 to CMC	x	x	x	x	x	-
Parity check address from CMC to CM	x	x	x	x	x	-
Parity check data between CM and control (non-SECDED mode only)	x	x	x	x	x	x
SECDED between LCME and LCME control	-	-	-	-	-	x
Parity check data between LCME and LCME control (non-SECDED mode only)	-	-	-	-	-	x
Parity check on PPS memory data	x	x	x	x	x	x
Parity check on PPU memory data	-	-	-	-	-	x

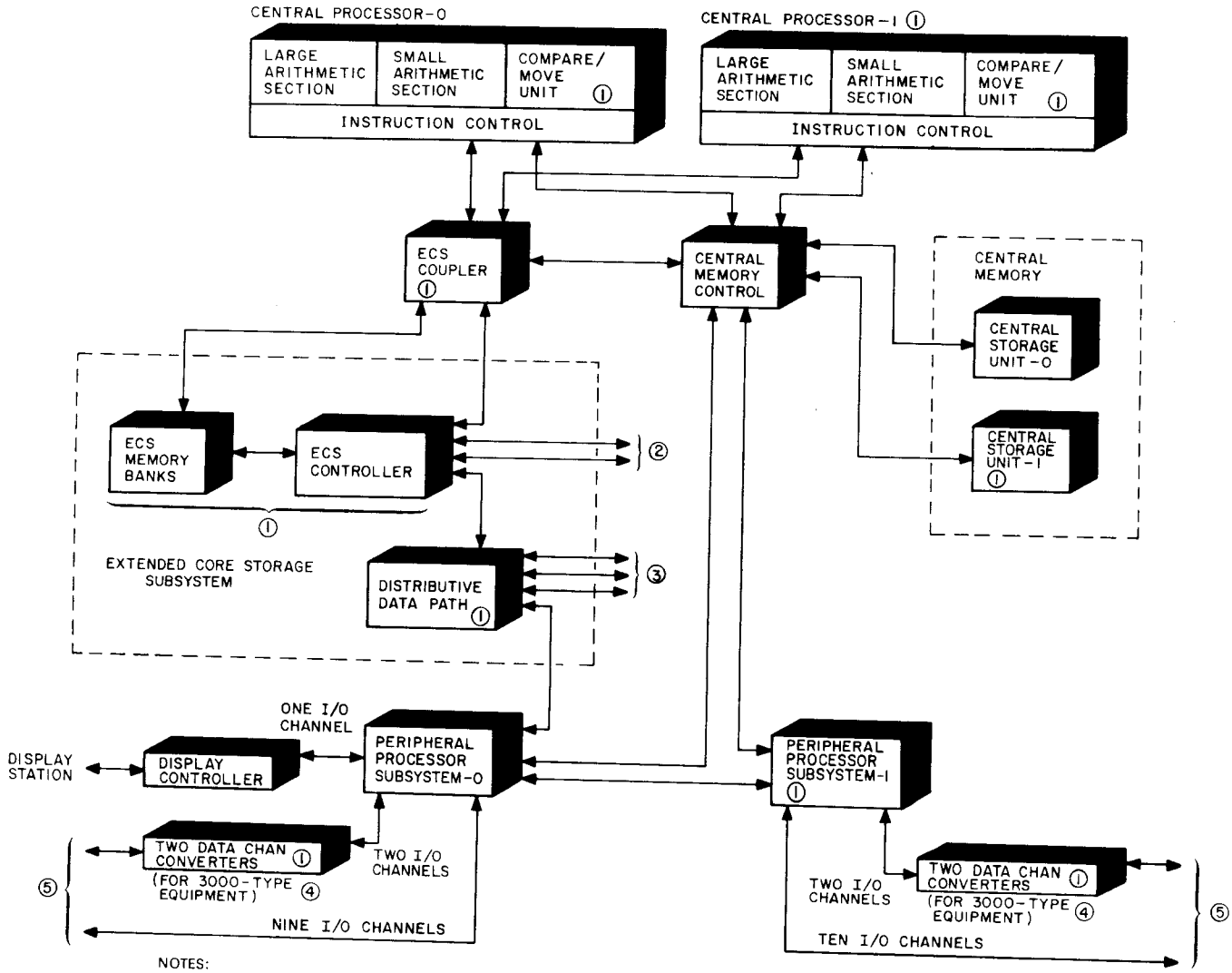
x Standard
- Not available

MODEL 171 SYSTEM

The model 171 basic computer system (figure 1-9) has a serial CP-0 with a serial CP-1 option. Each CP contains large and small arithmetic sections, instruction control, and a compare/move unit. The CPs communicate with each PPS and ECS, if installed, through CM. CM is under control of the CMC.

If the optional ECS is installed, it provides additional memory capabilities, short access times, and fast transfer rates to and from CM.

The PPS-0 performs all I/O operations and uses an instruction set separate from that of the CP to execute independent programs in each of 10 PPs. The PPs have individual memories and communicate with each other and any of 12 I/O channels. The PPs may be expanded from 10 to 14, 17, or 20 by adding PPS-1. This option expands the number of I/O channels from 12 to 24.



NOTES:

① OPTIONAL EQUIPMENT.

② TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS OR DISTRIBUTIVE DATA PATHS.

③ THREE PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

④ EXTERNAL DATA CHANNEL CONVERTERS MAY BE ADDED IN ADDITION TO THOSE IN PPS-0 AND PPS-1.

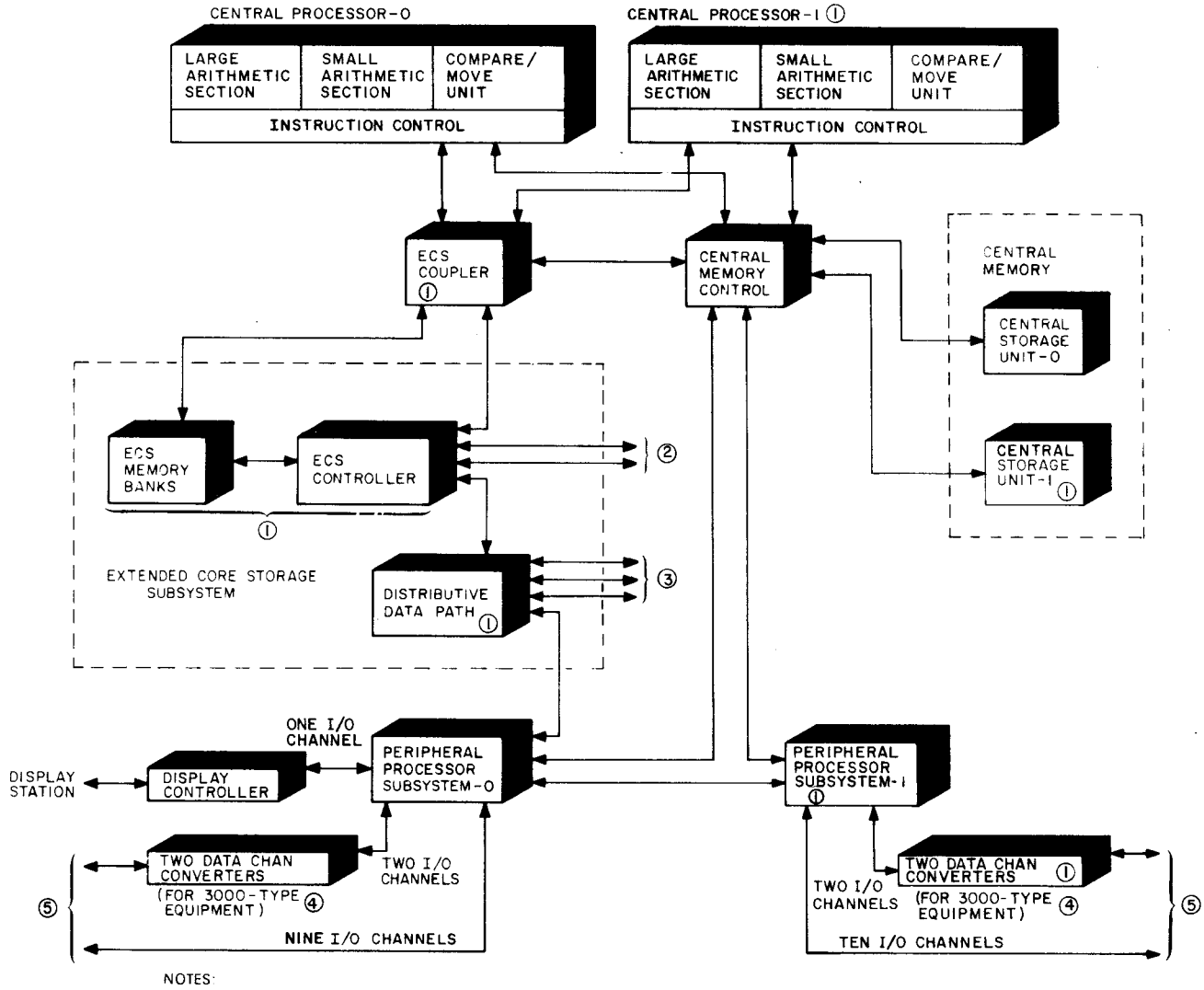
⑤ PERIPHERAL EQUIPMENT.

Figure 1-9. Model 171 Computer System

MODEL 172 SYSTEM

The model 172 basic computer system (figure 1-10) is functionally similar to model 171, except that the CP provides faster operation. The model 172 also has a second CP option. Basic equipment in model 172 includes a compare/move unit in the CP and two DCCs in PPS-0. This equipment is available only as options in model 171.

Original and later model 172 systems have some differences that result from development of additional equipment. Figure 1-10 represents the later model 172 system. The original model 172 basic memory has 32,768 words and requires two increments to reach the 65,536 words of the basic memory of later model 172 systems.



NOTES:

① OPTIONAL EQUIPMENT.

② TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS OR DISTRIBUTIVE DATA PATHS.

③ THREE PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

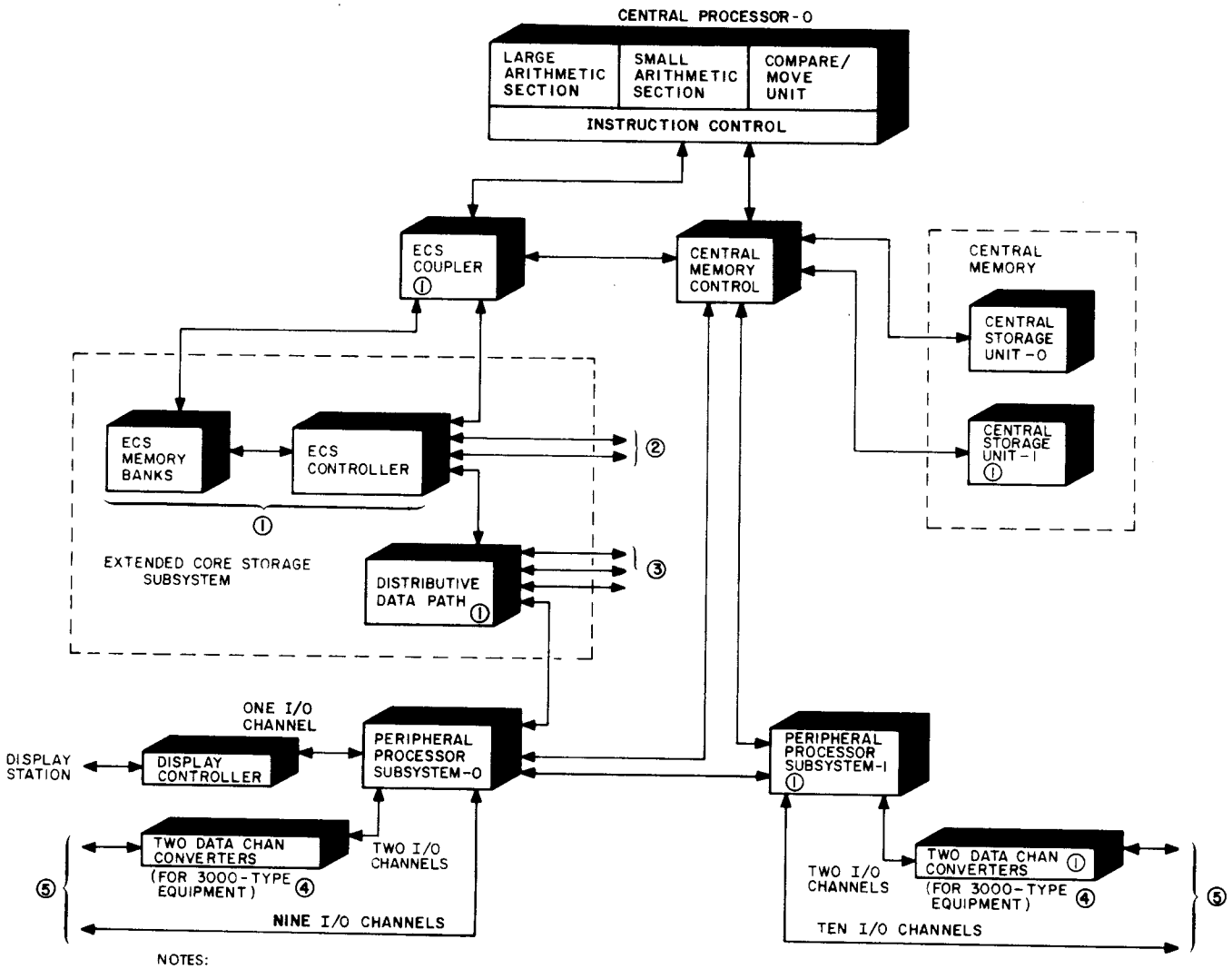
④ EXTERNAL DATA CHANNEL CONVERTERS MAY BE ADDED IN ADDITION TO THOSE IN PPS-0 AND PPS-1.

⑤ PERIPHERAL EQUIPMENT.

Figure 1-10. Model 172 Computer System

MODEL 173 SYSTEM

The model 173 basic computer system (figure 1-11) is functionally similar to model 172, except that the CP provides faster operation.



NOTES:

① OPTIONAL EQUIPMENT.

② TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS OR DISTRIBUTIVE DATA PATHS.

③ THREE PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

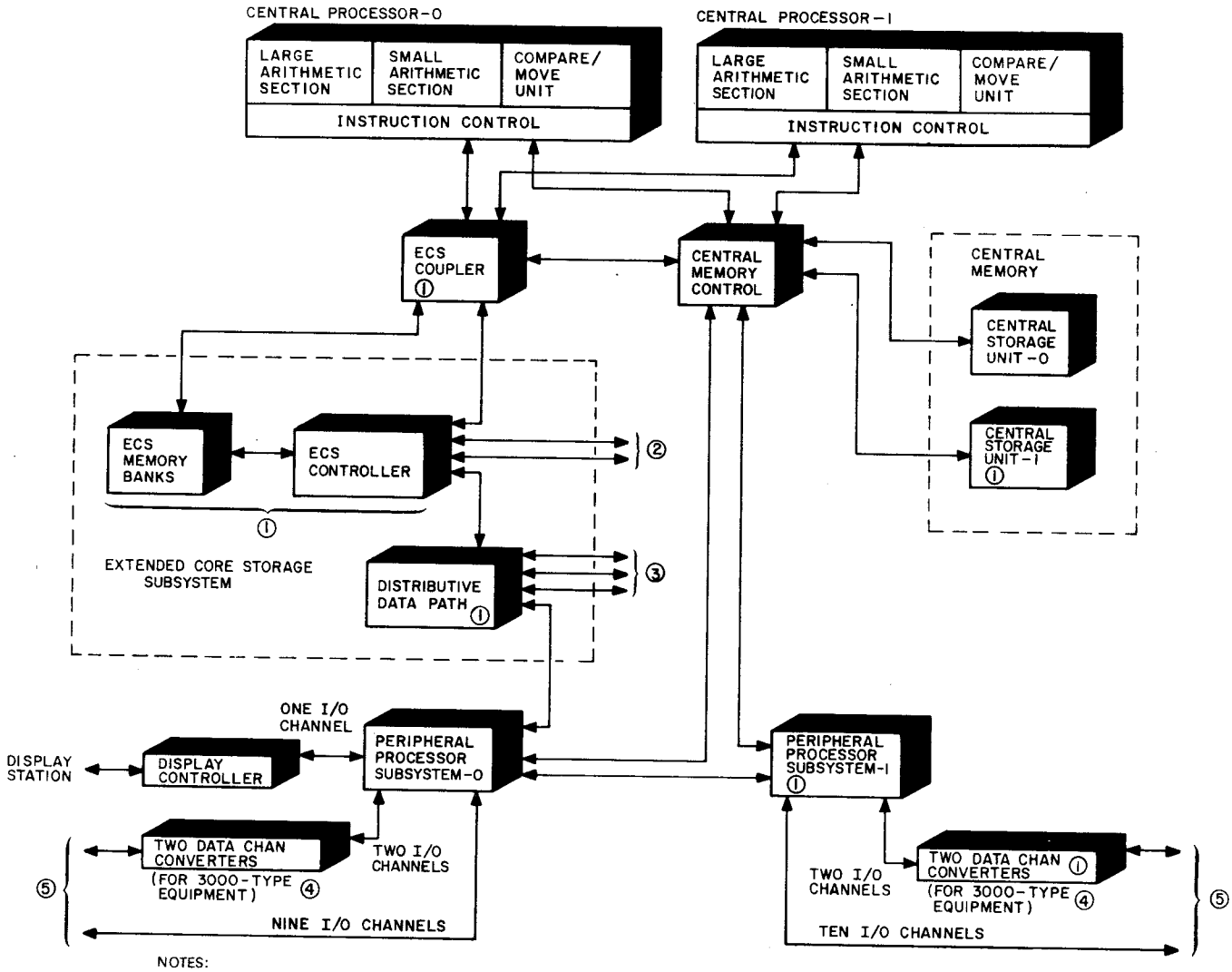
④ EXTERNAL DATA CHANNEL CONVERTERS MAY BE ADDED IN ADDITION TO THOSE IN PPS-0 AND PPS-1.

⑤ PERIPHERAL EQUIPMENT.

Figure 1-11. Model 173 Computer System

MODEL 174 SYSTEM

The model 174 basic computer system (figure 1-12) is functionally similar to model 173, except that the system provides faster operation. Model 174 differs basically from model 173 by having a second CP. The ECS, CM, PPS, and I/O options are the same as for model 173.



NOTES:

① OPTIONAL EQUIPMENT.

② TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS OR DISTRIBUTIVE DATA PATHS.

③ THREE PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

④ EXTERNAL DATA CHANNEL CONVERTERS MAY BE ADDED IN ADDITION TO THOSE IN PPS-0 AND PPS-1.

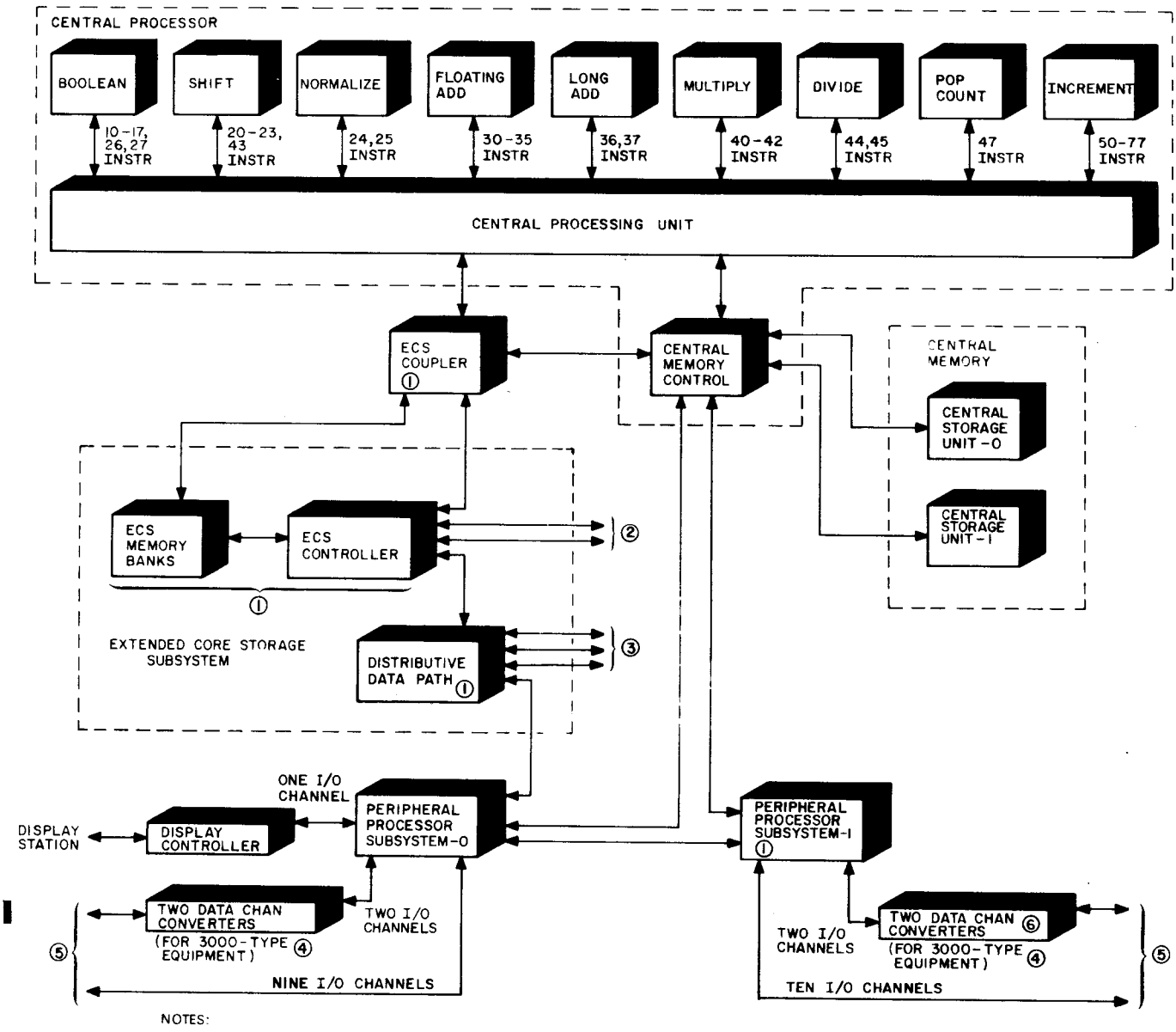
⑤ PERIPHERAL EQUIPMENT.

3AR13D

Figure 1-12. Model 174 Computer System

MODEL 175 SYSTEM

The model 175 basic computer system (figure 1-13) 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 that provide 16 independent banks of memory.



NOTES:

① OPTIONAL EQUIPMENT.

② TWO PORTS AVAILABLE FOR USE BY OTHER SYSTEMS OR DISTRIBUTIVE DATA PATHS.

③ THREE PORTS AVAILABLE AS OPTIONS FOR USE BY OTHER SYSTEMS.

④ EXTERNAL DATA CHANNEL CONVERTERS MAY BE ADDED IN ADDITION TO THOSE IN THE PPS.

⑤ PERIPHERAL EQUIPMENT.

⑥ OPTIONAL EQUIPMENT FOR MODELS 175A AND 175B. NOT AVAILABLE FOR MODEL 175C

Figure 1-13. Model 175 Computer System

MODEL 176 SYSTEM

The model 176 basic computer system (figure 1-14) is functionally similar to model 175 in the areas of the CP and PPS. Model 176 differs basically from model 175 in the use of LCME in the basic system instead of having an ECS option. The CM is still optionally expandable but does not have separate CSUs as in other models. The CM and LCME each contain their own control functions. Other major differences include the addition of four PPUs with up to nine more PPUs optionally available, an I/O multiplexer, and a logic scanner to permit PPS communication with the PPUs.

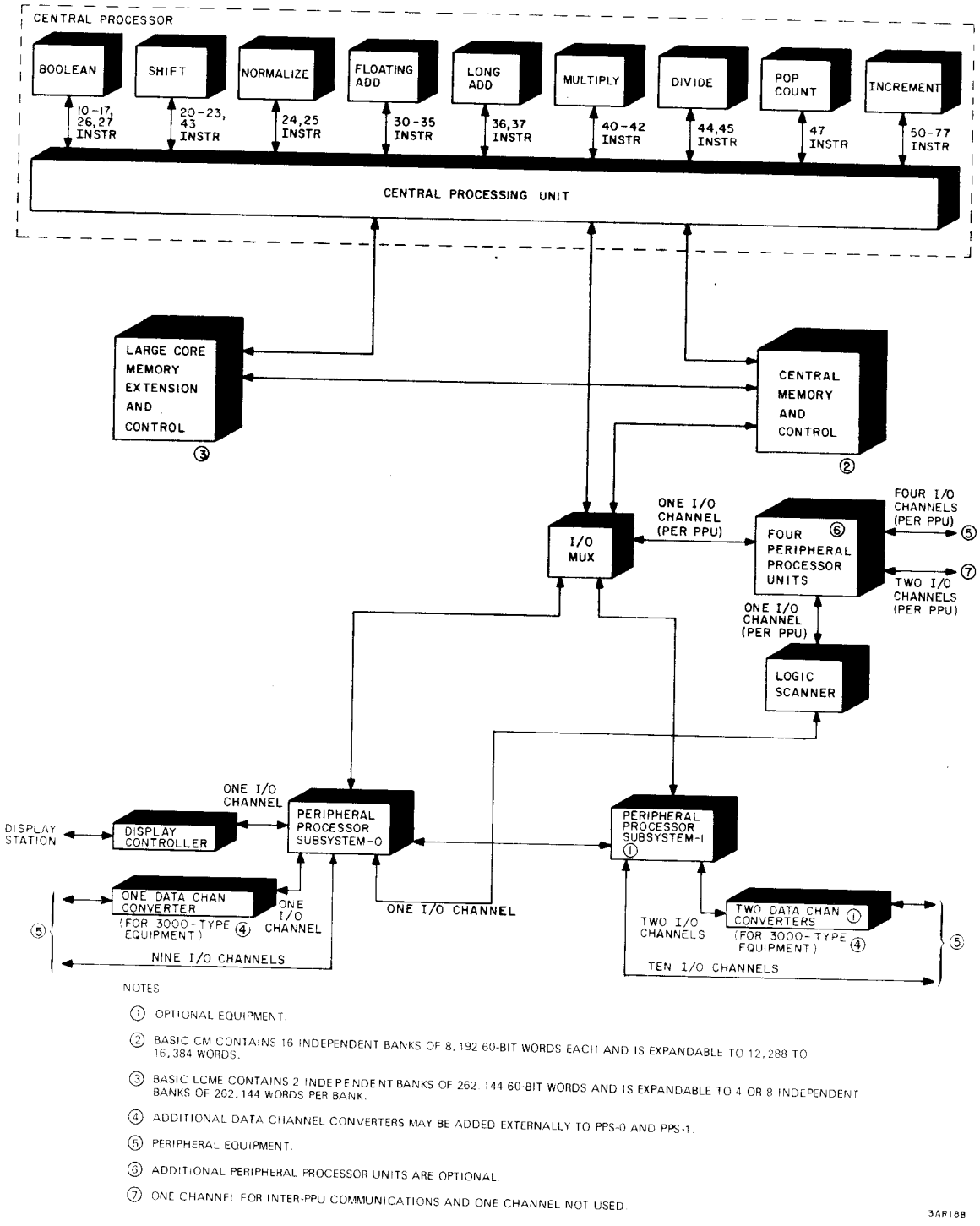


Figure 1-14. Model 176 Computer System

MAJOR SYSTEM COMPONENT DESCRIPTIONS

MAJOR SYSTEM COMPONENT DESCRIPTIONS

The following are the major system components.

- Central processor
- Central memory
- Extended core storage (optional) in models 171 through 175
- Large core memory extension in model 176
- Peripheral processor units in model 176
- Peripheral processor subsystem
- Display station
- Condensing unit(s)
- Power distribution unit in model 176

CENTRAL PROCESSOR — MODELS 171 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 retrieval, address preparation, memory protection, and data retrieval and storage. The control section acquires instructions from CM and decodes and executes them in a serial manner. Operating registers reduce storage accesses for operands used during the execution of an instruction. 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 use A0 primarily for indexing and A1 through A7 for CM operand addressing.
- Eight 18-bit B registers (B0 through B7) which are primarily indexing registers to control 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 CM word may contain up to 10 characters. 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 arithmetic sections consist of a large arithmetic section (used by instructions requiring manipulation of 60-bit operands) and a 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.

The CMC provides an interface between CM and five CM access ports (PPS-0, PPS-1, CPU-0, CPU-1, and ECS). The CMC primarily controls address and write data to CM and read data from CM. In addition, the CMC:

- Determines access priorities
- Increments addresses
- Checks and generates address and data parity
- Provides single-error correction double-error detection (SECDED)
- Performs breakpoint checks
- Controls CM refresh (models A and B only)
- Controls CM reconfiguration
- Controls exchange jumps

CENTRAL PROCESSOR — MODELS 175 AND 176

Models 175 and 176 differ from the other models by not having the compare/move capability and by performing parallel processing rather than serial processing within a single CP.

The model 175 and 176 CPs have basic similarities in their CPUs. Each CPU contains operating registers, support registers, and functional units. In addition, the model 175 CP contains a CMC. In model 176, the memory control function is part of the CM.

The operating 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; for model 176 only, these registers also input and output data and addresses for LCME
- Eight 18-bit A registers (A0 through A7) which use A0 primarily for indexing and A1 through A7 for addressing
- Eight 18-bit B registers (B0 through B7) which are primarily indexing registers to control program execution

The support registers support the operating registers during the execution of programs. These registers differ between models 175 and 176.

Model 175 support 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 (lower 6 bits are zeros)
- Field length for ECS (FLE) register, 24 bits (lower 6 bits are zeros)
- Monitor address (MA) register, 18 bits

Model 176 support registers are:

- Program address (P) register, 18 bits
- Reference address for CM (RAS) register, 18 bits
- Field length for CM (FLS) register, 18 bits
- Program status designator (PSD) register, 18 bits
- Reference address for LCME (RAL) register, 22 bits
- Field length for LCME (FLL) register, 22 bits
- Normal exit address (NEA) register, 18 bits
- Error exit address (EEA) register, 18 bits

Instruction control 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 rereading instructions from 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 which have a value of one in a 60-bit operand
- Increment unit which forms the one's 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.

In model 175, the CMC controls the flow of data between CM and the requesting elements of the system. In addition, the CMC:

- Determines priorities
- Increments addresses
- Checks and generates address and data parity
- Provides single-error correction double-error detection (SECDED)
- Performs breakpoint checks
- Controls CM refresh (models A and B only)
- Controls CM reconfiguration

CENTRAL MEMORY — MODELS 171 THROUGH 176

The CM in models 171 through 175 is a metal oxide semiconductor (MOS) memory. The CM in model 176 is a bipolar semiconductor memory. Each of the basic CM sizes is field-upgradable to 262,144 words.

Words in the CMs contain 60 data bits and 8 SECDED code bits.

EXTENDED CORE STORAGE (OPTIONAL) — MODELS 171 THROUGH 175

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

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

The ECS controller regulates the computer system access to the ECS memory bays through four available access ports. One access port connects to the ECS coupler. The other ports may connect to other systems or optional DDPs. Each access port carries 60 data bits plus 1 parity bit and control signals. Eight 60-bit data words plus eight parity bits comprise an ECS record. The ECS controller performs time-sharing of ECS records in the four access ports during ECS data transfers. The ECS controller interfaces from one to four ECS memory bays and carries 60 data bits plus 1 parity bit. Depending upon the controller used, the controller transfers the parity bit that accompanies the data from the computer system to the ECS memory bays or generates a parity bit for the ECS data.

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 2,097,152 words (16 banks). A cabinet, termed bay, holds up to four memory banks. Each ECS bank address stores one ECS record. References of one 60-bit word are possible.

The DDP 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.

Each ECS requires an ECS coupler which mounts within the mainframe cabinet. The coupler interfaces the mainframe with the ECS processing and monitoring data and control between the systems. The coupler:

- Receives the 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

Additional information for the ECS and ECS coupler is in manuals listed in the system publication index in the preface of this manual.

LARGE CORE MEMORY EXTENSION — MODEL 176

LCME provides additional storage for data that is not immediately needed by the CPU. The data transfers through a bidirectional high-speed data path. Data may also be transferred one word at a time to or from the X operating registers. However, programs cannot execute directly out of LCME.

In a basic system configuration, LCME contains 512,288 words, expandable with system options to 2,097,152 72-bit words. Each word includes 60 data bits, 8 error correction bits, and 2 complement control bits.

PERIPHERAL PROCESSOR UNITS — MODEL 176

The PPU is a computer with an independently stored program. The system contains from 4 to 13 PPUs depending on the number installed as options. The PPUs have 4096 words (12 bits plus 1 parity bit per word) of coincident current memory organized into two independent banks of 2048 words. The PPUs share access to CM with the PPS through I/O multiplexer channels. Each PPU operates independently with separate hardware for performing arithmetic, logical, and I/O operations.

PERIPHERAL PROCESSOR SUBSYSTEM — MODELS 171 THROUGH 176

The PPS consists of 10 logically independent computers in PPS-0 and 4, 7, or 10 of the computers in PPS-1, when installed as options. 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 operate 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 system 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.

Model 171 through 176 mainframes are expandable 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. All I/O channels are accessed by all PPs.

DISPLAY STATION — MODELS 171 THROUGH 176

The 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 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 in the preface of this manual for further display station information.)

CONDENSING UNIT(S) — MODELS 171 THROUGH 176

One or more condensing units circulate cooling refrigerant to cold bars and plates for the conduction cooling of the logic and memory paks and logic and memory modules in a system. For models 171 through 174, one 3-ton condensing unit mounts in and cools each bay. For model 175, one 10-ton condensing unit is in a stand-alone cabinet and cools the entire system. For model 176, two 10-ton condensing units in stand-alone cabinets cool the basic system. System options permit a third 10-ton condensing unit.

POWER DISTRIBUTION UNIT — MODEL 176

The PDU distributes 400-Hz power to the dc power supplies located in the mainframe. It also contains a warning system that monitors logic chassis temperature, room dew-point temperature, and condensing unit condition. A warning panel in the PDU contains relay circuits that activate a horn and automatically shut off computer power when the cooling system malfunctions.

FUNCTIONAL DESCRIPTIONS

This section provides functional descriptions of the system mainframe parts shown in the block diagrams in section 1. These parts consist of:

- Central processor (CP) in models 171 through 174
- Central processor in models 175 and 176
- Central memory control (CMC) in models 171 through 175
- Central memory (CM) in models 171 through 175
- Central memory in model 176
- Large core memory extension (LCME) in model 176
- Input/output multiplexer (MUX) in model 176
- Logic scanner in model 176
- Data channel converter (DCC)
- Display controller
- Peripheral processor units (PPUs) in model 176
- Peripheral processor subsystem (PPS)

Functional descriptions for the system display station, condensing units, and extended core storage (ECS) are in respective manuals listed in the system publication index in the preface of this manual.

Functional differences exist among all of the CDC CYBER 170 models. The main differences exist between the serial program processing of the CPs of models 171 through 174 and the parallel program processing of the CPs of models 175 and 176.

CENTRAL PROCESSOR - MODELS 171 THROUGH 174

CENTRAL PROCESSOR — MODELS 171 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. The compare/move units are optional on model 171.

ARITHMETIC SECTIONS — MODELS 171 THROUGH 174

The large and small arithmetic sections form a unified arithmetic unit. Instructions use the large section for 60-bit operand manipulation and the small section for 18-bit operand and exponent manipulation. The large arithmetic section contains a 108-bit adder, shift network, normalize network, and shift counter. 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 — MODELS 171 THROUGH 174

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 models 175 and 176 and are repeated in the description of the model 175 and 176 CPs to provide a continuous system description.

Operating Registers — Models 171 through 174

The operating registers consist of operand (X), address (A), and index (B) registers. These registers minimize memory references for arithmetic operands and results.

X Registers

The CP contains eight 60-bit X registers, X0 through X7. The X0 register is used in the compare instructions to indicate if two fields of characters are equal. If the system includes ECS, the X0 register provides the relative starting address in a block copy operation. The X0 register also provides the instruction information during a flag register operation.

The X1 through X7 registers are primarily data handling registers for computation with X1 through X5 used to input data from CM and X6 and X7 used to transmit data to CM.

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

A Registers

The CP contains eight 18-bit A registers, A0 through A7. The A0 register serves as an intermediate register for the user's discretion. If the system includes ECS, the A0 register provides the relative CM starting address. The A0 register is also used for the collation table address. The register is not used in an ECS flag operation.

The A1 through A7 registers 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 address and transmits the CM word to the corresponding register (X1 through X5). Similarly, placing a quantity into the A6 or A7 register causes the word in the corresponding X6 or X7 register to be written into that relative address of CM.

B Registers

The CP contains eight 18-bit B registers, B0 through B7. These registers are primarily indexing registers to control program execution. Program loop counts may also 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 resultant exponent for the unpack, the operand exponent for the pack, and the resultant shift count from a normalize. The B0 register always contains positive zero which can be used as an operand. This register cannot hold results from instructions.

Support Registers — Models 171 through 174

Seven support registers assist the operating registers during the execution of programs. The support registers load 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 completes, the data in the support registers is sent back to CM through an exchange jump.

P Register

The 18-bit program address (P) register loads from CM during the first word of an exchange sequence and 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

The 18-bit CM reference address (RAC) register loads from CM during the second word of an exchange sequence. An absolute CM address forms by adding RAC to a relative address determined by

the instruction. 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

The 18-bit CM field length (FLC) register loads from CM during the third word of an exchange sequence. The FLC register defines 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 Exit Mode/Error Response under Central Processor in section 5.)

EM Register

The six-bit exit mode (EM) register loads 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	Infinite operand
50	Indefinite operand
57	Parity error on ECS flag register operation
58	Central processor unit (CPU) to CMC address or data parity error or CPU to CMC to CM address parity error
59	CMC to CPU data parity error or double error

RAE Register

The 21-bit ECS reference address (RAE) register loads from CM during the fifth word of an exchange sequence. The lower six bits of this register are always zero. An absolute ECS address forms by adding RAE to the relative address which is determined by the instruction.

FLE Register

The 24-bit ECS field length (FLE) register loads from CM during the sixth word of an exchange sequence. The lower six bits of this register are always zero. The FLE register defines the size of the field in ECS for the program in execution. Relative ECS addresses are compared with FLE. (For further information, refer to Exit Mode/Error Response under Central Processor in section 5.)

MA Register

The 18-bit monitor address (MA) register loads 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 (013) instruction with the monitor flag clear or when honoring a monitor exchange jump to MA (262x) instruction with the monitor flag clear.

Instruction Control Sequences — Models 171 through 174

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 CP Instruction Descriptions in section 4.)

Boolean Sequence

The boolean sequence controls instructions that require bit-by-bit data 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 $(\overline{Xk})$ to Xi
16	Logical sum of (Xj) and $(\overline{Xk})$ to Xi
17	Logical difference of (Xj) and $(\overline{Xk})$ to Xi

The instructions requiring transmissive operations are:

10 Transmit (Xj) to Xi

14 Transmit ($\overline{Xk}$) 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 Floating-Divide Sequence

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

The multiply instructions are:

40 Floating product of (Xj) and (Xk) to Xi

41 Round 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) by (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 one's complement addition and subtraction of 18-bit fixed-point operands for increment instructions 50 through 77. The sequence also controls the 60-bit one's 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) minus (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 are 60-bit instructions that use six support registers for source and result field CM addresses and character position offsets. The support registers load from the 60-bit instruction word. The compare/move instructions are:

464 Move indirect (Bj)+ K

465 Move direct

466 Compare collated

467 Compare uncollated

The support registers are:

- An 18-bit K1 register that specifies which relative CM address word contains the first character of the source data field.
- An 18-bit K2 register that specifies which relative CM address word contains the first character of the result field.
- A 4-bit C1 register that specifies the character position or offset of the first CM word of the source field.
- A 4-bit C2 register that specifies the character position or offset of the first CM word of the result field.
- Two 14-bit L registers (LA and LC) that specify the number of characters in the data field. The LA register is associated with K1, and the LC register is associated with K2. Instruction 464 uses 14 register bits. Instructions 465, 466, and 467 use only the lower eight register bits.

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 content of an exchange jump package.

The CMC always initiates the exchange sequence from a CP or peripheral processor (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 (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 three instructions.

00 Error exit to MA or program stop

010 Return jump to K

013 Central exchange jump to (Bj) + K or (MA)

CENTRAL PROCESSOR - MODELS 175 AND 176

CENTRAL PROCESSOR — MODELS 175 AND 176

The CP consists of a central processing unit (CPU) and nine functional units. The model 175 CP includes a CMC. In model 176, the control for CM is part of CM, and the control for LCME is part of LCME.

CENTRAL PROCESSING UNIT — MODELS 175 AND 176

The CPU consists of instruction control, 24 operating registers, and 7 or 8 support registers respectively, for models 175 and 176. The CPU includes the registers and control logic to direct the arithmetic operations and provide interface between the functional units and CMC in model 175. In addition to instruction execution, the CPU performs instruction fetching, address preparation, memory protection, and data fetching and storing. Figure 2-1 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 useable state for later reentry into the CPU. (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- 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.

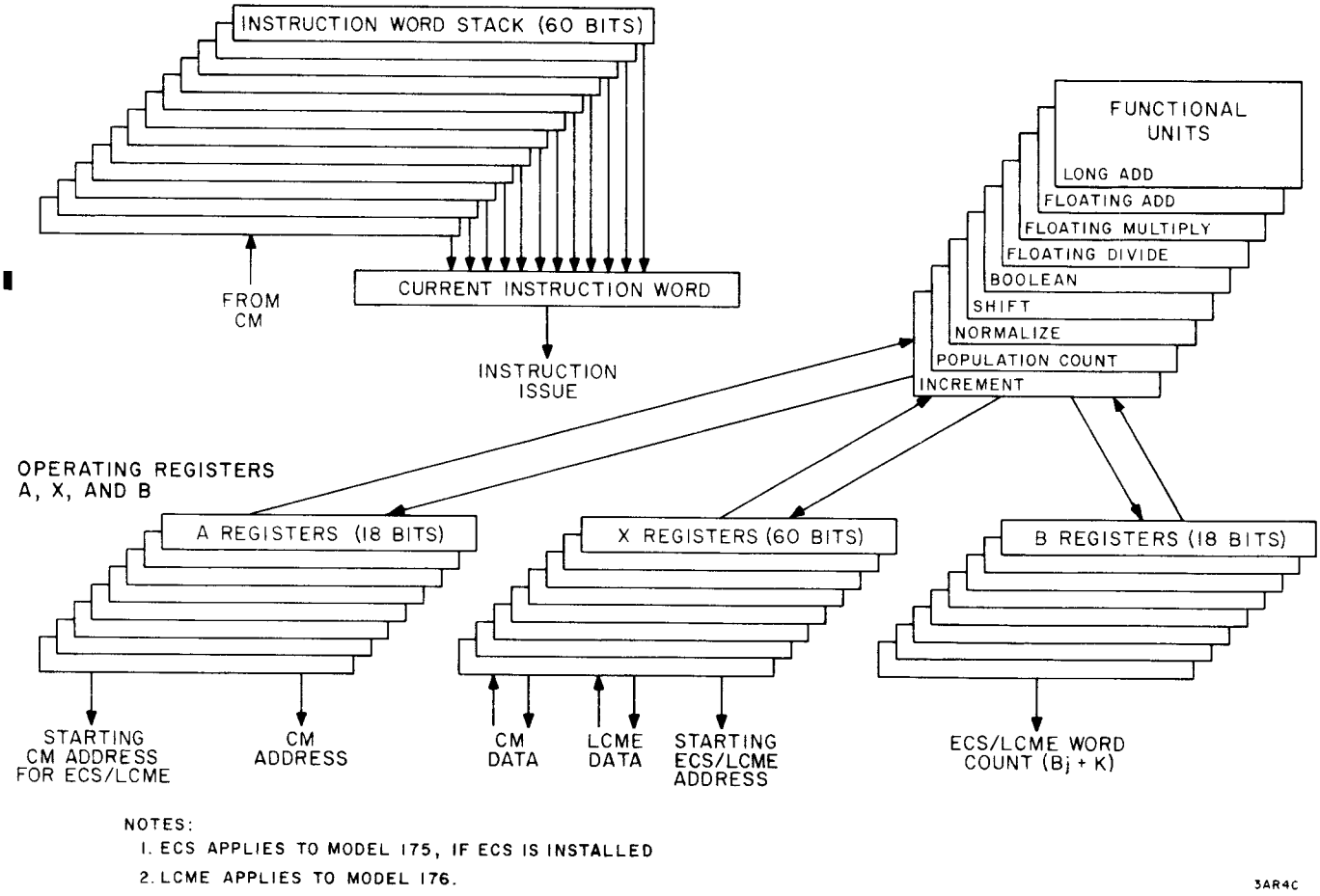


Figure 2-1. Models 175 and 176 CPU Information Flow

The following operating and support register descriptions, although identical to those for models 171 through 174, are repeated here to provide a continuous system description.

Operating Registers — Models 175 and 176

The operating registers consist of operand (X), address (A), and index (B) registers. These registers minimize memory references for arithmetic operands and results.

X Registers

The CP contains eight 60-bit X registers, X0 through X7. The X0 register provides the relative starting address in ECS/LCME for a block copy operation. The X0 register also provides the instruction information during a flag register operation for a system with ECS.

The X1 through X7 registers are primarily data handling registers for computation with X1 through X5 used to input data from CM and X6 and X7 used to transmit data to CM. All 60-bit operands involved in computation must originate and terminate in X1 through X7. The X registers are also operand registers when referencing single words from LCME.

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

A Registers

The CP contains eight 18-bit A registers, A0 through A7. The A0 register serves as an intermediate register for the user's discretion. The A0 register provides the relative CM starting address for ECS/LCME operations. The register is not used in a flag operation.

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

B Registers

The CPU contains eight 18-bit B registers, B0 through B7. These registers are primarily indexing registers to control program execution. Program loop counts may also be incremented or decremented in these registers. Additionally, the B registers hold the Bj portion of the ECS or LCME word count, Bj portion of the exchange jump address, channel number for I/O instructions and jump index for the long jump (02) instruction.

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. The B0 register always contains positive zero which can be used as an operand. This register cannot hold results from instructions.

Support Registers — Model 175

The support registers assist the operating registers during the execution of programs. The support registers load 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 completes, the data in these registers is sent back to CM.

P Register

The 18-bit P register loads from the first word of an exchange sequence and 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 advances one address ahead of the instruction in progress, a P buffer register holds the current program execution address. This buffered address is used for the PPS read program address instruction (27X). The contents of the P register advances to the next program step as follows:

1. The P register advances by one when an instruction word is sent to the CIW register.
2. The P register sets 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 sets to the address specified in the exchange package.

RAC Register

The 18-bit CM reference address (RAC) register loads from CM during the second word of an exchange sequence. An absolute CM address forms by adding RAC to a relative address determined by the instruction. 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

The 18-bit CM field length (FLC) register loads from CM during the third word of an exchange sequence. The FLC register defines 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 Exit Mode/Error Response under Central Processor in section 5.)

EM Register

The six-bit exit mode (EM) register loads 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	Infinite operand
50	Indefinite operand
57	Parity error on ECS flag register operation
58	CPU to CMC address or data parity error or CPU to CMC to CM address parity error
59	CMC to CPU data parity error or double error

RAE Register

The 21-bit ECS reference address (RAE) register loads from CM during the fifth word of an exchange sequence. An absolute ECS address forms by adding RAE to the relative address determined by the instruction.

FLE Register

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

MA Register

The 18-bit monitor address (MA) register loads 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 (013) instruction with the monitor flag clear or when honoring a monitor exchange jump to MA (262x) instruction with the monitor flag clear.

Support Registers — Model 176

The support registers assist the operating registers during the execution of programs. The support registers load from CM during an exchange sequence (refer to Exchange Jump in section 5). With the exception of the P register and the PSD condition designators, the contents of the support registers cannot be altered during the execution interval of an exchange package. When the execution interval completes, the data in these registers is sent back to CM.

P Register

The 18-bit P register loads from the first word of an exchange sequence and 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 advances one address ahead of the instruction in progress, a P buffer register holds the current program execution address. This buffered address is used for the PPS read program address (27X) instruction. The contents of P register advances to the next program step as follows:

1. The P register advances by one when an instruction word is sent to the CIW register.
2. The P register sets 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 sets to the address specified in the exchange package.

RAS Register

The 18-bit reference address for CM (RAS) register loads from CM during the second word of an exchange sequence. An absolute CM address is formed by adding RAS to the relative address that is determined by the instruction. CM references from the MUX are absolute addresses and therefore are not added to RAS.

FLS Register

The 18-bit field length for CM (FLS) register loads from CM during the third word of an exchange sequence. Relative CM addresses are compared with FLS. If a relative CM address equals or exceeds FLS, the CM block range of CM direct range condition flag sets in the PSD register.

PSD Register

The program status designator (PSD) register (figure 2-2) is a collection of 18 program status flags. Six flags are mode designators, and 12 flags are condition designators.

The PSD register loads from the exchange package during an exchange jump sequence. All 18 flag bits enter in the register at this time. The six mode designators remain unaltered throughout the execution interval for the exchange package. The 12 condition designators may be set by conditions that occur during the execution interval. All flags store in the CM exchange package at the end of the execution interval.

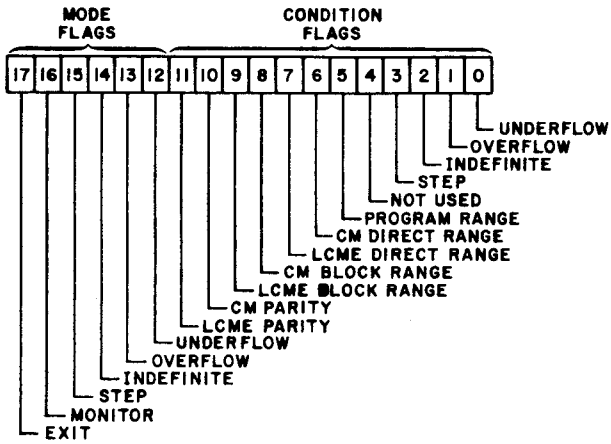


Figure 2-2. PSD Register Flag Bit Arrangement

Mode Flags – The upper six bits (12 through 17) in the PSD register are mode flags. These flags remain unaltered throughout the execution interval for the exchange package.

The exit mode flag (bit 17) determines the initial CM address for the exchange package during execution of an exchange exit (013) instruction. If this flag is set, the exchange package address is K plus the content of Bj plus the content of RAS. If clear, the address is the content of NEA.

The monitor mode flag (bit 16) determines when an I/O interrupt request or PPS exchange is honored. If this flag sets, the current program continues until execution is complete. If an I/O interrupt request occurs during this time, it is not honored until the end of the current program. If this flag clears, an I/O interrupt is honored immediately. The monitor mode flag also controls execution of the reset buffer instructions, 0160 and 0170. If this flag sets, the instruction executes as described. If this flag clears, the instruction executes as a pass instruction.

If the step mode flag (bit 15) sets and the first instruction of the current exchange interval issues, the step condition flag (bit 3) sets. The step condition flag then terminates the execution interval after the last instruction in the CIW register issues.

The indefinite mode flag (bit 14) enables interruption of the current exchange interval when an indefinite floating-point result occurs. If this flag and the indefinite condition flag (bit 2) set, the execution interval terminates after the last instruction in the CIW register issues.

The overflow mode flag (bit 13) enables interruption of the current exchange interval when an overflow of the floating-point range occurs. If this flag and the overflow condition flag (bit 1) set, the execution interval terminates after the last instruction in the CIW register issues.

The underflow mode flag (bit 12) enables interruption of the current exchange interval when an underflow of the floating-point range occurs. If this flag and the underflow condition flag (bit 0) set, the execution interval terminates after the last instruction in the CIW register issues.

Condition Flags – The lower 12 bits (0 through 11) in the PSD register are condition flags. These flags may set from the exchange package or from conditions that may occur during the execution interval. When this occurs, the execution interval for the exchange package terminates after the last instruction in the CIW register issues.

The LCME error flag (bit 11) sets when a double-bit SECDED or parity error exists in a word read from LCME.

The meaning of the CM error flag (bit 10) depends on the mode conditions controlled by the status and control register.

- Single-error correction double-error detection (SECDED) mode

  This is the normal operating mode. It is active whenever the parity mode bit from the status and control register is not present. Bit 10 sets when a double error is detected in a word read from CM. Single errors are automatically corrected, and bit 10 does not set.

- Inhibit log single-bit error (SBE) mode

  Similar to SECDED mode except that single-bit errors are logged by the status and control register. Single errors are corrected, and bit 10 does not set. Double-error detection sets the bit 10 flag.

- Maintenance mode

  Bit 10 sets when either a single or double error is detected in a word read from CM. Single errors are automatically corrected as in all SECDED modes. The log SBE mode must be active whenever maintenance mode is active.

- Parity mode

  The SECDED feature is disabled. Conventional parity checking takes place as in CM. Bit 10 sets when a parity error occurs in a word read from CM. The log SBE mode must be active whenever parity mode is active.

When a block copy instruction issues and causes an LCME reference to an address which equals or exceeds the lowest-order 22 bits of the register for the LCME field length (FLL), the LCME block range condition flag (bit 9) sets.

When a block copy instruction issues and causes a CM reference to an address which equals or exceeds the content of the register for the CM field length (FLS), the CM block range condition flag (bit 8) sets.

When a direct read or write LCME instruction issues and causes an LCME reference to an address which equals or exceeds the lowest-order 22 bits of the FLL register, the LCME direct range condition flag (bit 7) sets.

When an increment (50 through 57) instruction issues and causes a CM reference to an address equal to or greater than FLS or when P is equal to or greater than FLS, the CM direct range condition flag (bit 6) sets.

The program range condition flag (bit 5) sets whenever P equals zero or an error exit (00) instruction issues. When this flag sets by a 00 instruction, execution terminates immediately.

The step condition flag (bit 3) sets whenever the step mode flag (bit 15) sets and a new word enters the CIW register.

The indefinite condition flag (bit 2) sets when one of the floating-point functional units generates an indefinite result. The execution interval does not terminate unless the indefinite mode flag (bit 14) also sets.

The overflow condition flag (bit 1) sets when an overflow of the floating-point range occurs in the result from a functional unit. The execution interval does not terminate unless the overflow mode flag (bit 13) also sets.

The underflow condition flag (bit 0) sets when an underflow of the floating-point range occurs in the result from a functional unit. The execution interval does not terminate unless the underflow mode flag (bit 12) also sets.

RAL Register

The 22-bit reference address for LCME (RAL) register loads from CM during the fifth word of an exchange sequence. An absolute LCME address forms by adding the lowest-order 22 bits of RAL to the relative address determined by the instruction.

FLL Register

The 22-bit field length for LCME (FLL) register loads from CM during the sixth word of an exchange sequence. Relative LCME addresses are compared with FLL. If a relative LCME address equals or exceeds the lowest-order 22 bits of FLL, the LCME block range or LCME direct range condition flag sets in the PSD register.

NEA Register

The 24-bit normal exit address (NEA) register loads from CM during the seventh word of an exchange sequence. This register is used during an exchange exit (013) instruction when the exit mode flag in the PSD register clears. When this occurs, the current program terminates with an exchange sequence. The absolute CM address for the new exchange package is in the lowest-order 18 bits of NEA.

EEA Register

The 24-bit error exit address (EEA) register loads from CM during the eighth word of an exchange sequence. This register is used whenever an error exit occurs during the execution interval for an exchange package. When this occurs, the lowest-order 17 bits of EEA comprise the absolute address in CM for the exchange package that terminates the program.

Instruction Control Sequences — Models 175 and 176

The main instruction control components include an IWS, instruction address stack (IAS), and CIW. The instruction control reads 60-bit instruction words from CM and issues them to the CP functional units for execution in 15-, 30-, or 60-bit instruction groups for model 175 and in 15- or 30-bit instruction groups for model 176. The instruction control also performs instruction translation and control of the exchange, ECS/LCME block copy, LCME direct reads and writes, 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 fills 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.

The 12 IWS registers are individually identified by rank. The rank 1 register contains the oldest data. If the IWS in model 175 contains sequential program instruction words, the rank 1 register corresponds to the lowest CM address in the IAS.

Under certain conditions the IWS in model 175 or model 176 may be voided. Voiding the stack means that the IWS is not accessible, and the IAS clears. New instructions must then be read from CM into the IWS and IAS.

In model 175, voiding the stack results from an exchange jump, return jump, jump to Bi plus K (02 instruction) or a branch (03 through 07 instructions) to a location not in the stack. The stack always contains a sequential code.

In model 176, voiding the stack results from an exchange or return jump. This stack can contain a nonsequential code or duplicate entries.

The IWS shifts to accommodate a new word arriving from CM. New information arriving from CM enters in rank 12. Ranks 11 through 1 clear and enter with information from the next highest-order-rank. The information in rank 1 discards.

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 relative 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 in model 175 permits disabling of 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 load 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 then left shift in the CIW register by either 15 or 30 bits, depending upon the instruction format.

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 memory control or a PP multiplexer interrupt request.

ECS/LCME Block Copy Sequence

The ECS/LCME block copy sequence controls the transfer of data between CM and ECS/LCME. 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/RAS reference address. The starting address for ECS/LCME is obtained from the X0 register plus the RAE/RAL reference address. The ECS/LCME block copy instructions are:

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

In model 176, (Bj + K) cannot exceed 1777_8.

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 model 175 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.

00	Error exit to MA or program stop for model 175 and EEA for model 176
010	Return jump to K
013	Central exchange jump to (Bj) + K or exchange exit to NEA

FUNCTIONAL UNITS — MODELS 175 AND 176

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 issues.

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 executes. No segmentation is 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 that require bit-by-bit data 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 $(\overline{Xk})$ to Xi
16	Logical sum of (Xj) and $(\overline{Xk})$ to Xi
17	Logical difference of (Xj) and $(\overline{Xk})$ to Xi

The instructions requiring transmissive operations are:

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

Shift Unit

The shift unit executes instructions that require shifting the entire 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

Normalize Unit

The normalize unit executes instructions that require rearranging operands in floating-point format. The unit left shifts the coefficient so that the most significant bit shifts into the highest-order bit position of the coefficient. The exponent adjusts by subtracting the shift count. The normalize 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 that require adding operands in floating-point format. 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

Long Add Unit

The long add unit executes instructions that require integer addition of two 60-bit operands. The long add 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 that require multiplication of two operands in floating-point format. The multiply instructions are:

40 Floating product of (Xj) and (Xk) to Xi

41 Round 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 that require division of two operands in floating-point format. The divide 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 that requires counting the number of one bits in a 60-bit operand. The population-count instruction is:

47 Population count of (Xk) to Xi

Increment Unit

The increment unit executes instructions 50 through 77 that require arithmetic operations on two 18-bit operands. During the 50 through 57 instructions, the result transmits to an A register. The same result plus RAC is sent to CM for the increment read or write address. The two operations perform independently and in parallel with each other. During 60 through 67 instructions, the result transmits to a B register. During 70 through 77 instructions, the result transmits 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 CONTROL - MODELS 171 THROUGH 175

CENTRAL MEMORY CONTROL - MODELS 171 THROUGH 175

The CMC controls the flow of data between CM and the requesting system components. In models 171 through 174, CMC is separated from the CP chassis. In model 175, CMC is part of the CP chassis. This 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-0, CP-1, PPS-0, PPS-1, and ECS (models 171, 172, and 174)

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

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

- Resolves CM bank conflicts including bank busy and reservations
- 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-0 (models 171, 172, and 174)

  CM and CP (model 175)

- Parity checks addresses from:

  CP-0, CP-1, PPS-0, and PPS-1 (models 171, 172, and 174)

  CP, PPS-0, and PPS-1 (model 173)

  PPS-0 and PPS-1 (model 175)

- Parity checks data from:

  CP-0, CP-1, PPS-0, PPS-1, and ECS (models 171, 172, and 174)

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

  PPS-0, PPS-1, and ECS (model 175)

- Generates parity on data to:

  CP-0, CP-1, PPS-0, PPS-1, and ECS (models 171, 172, and 174)

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

  PPS-0, PPS-1, and ECS (model 175)

- Generates parity for address to CM (models 171 through 175)
- Generates and checks data parity for CM in parity mode
- Breakpoint checks
- Controls CM refresh (models A and B only)
- Controls CM reconfiguration
- Performs SECDED on each memory word in SECDED mode, described in SECDED in this section

REFERENCE PRIORITIES

When a CM reference is initiated in models 171 through 174, the address goes to all CM banks. Only the bank selected 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) does not issue until CM accepts the reference from the address buffer. In models with 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 the bank selected 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, are accepted by CM. This address backup condition in SAS does not occur when doing an ECS transfer.

In models 171 through 175, all addresses presented to CMC process in the order in which they are received. CMC requests received simultaneously are given a priority that determines which address is allowed access first. These priorities are:

- CP exchange jump sequence (CEJ) request for CP-0, then CP-1 if CP-1 is present
- Exchange request from PPS-0, then PPS-1 if PPS-1 is present
- ECS block transfer request for CP-0, then CP-1 if CP-1 is present
- Read/write request from PPS-0, then PPS-1 if PPS-1 is present
- Read, write, and read next instruction (RNI) requests from CP-0, then CP-1 if CP-1 is present

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 are free when needed for ECS.

SECDED MODE

SECDED is a normal CMC operating mode that permits unimpeded computer operation despite a single-bit CM failure. The SECDED mode is manually selected with a switch that also allows the CMC to be set in a non-SECDED or parity mode. The SECDED mode is accomplished by a SECDED network which corrects single data errors from CM. 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 2-3) 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 bits of 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 64 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 generates. The single-error correction network automatically corrects the incorrect bit. If two data bits fail, a syndrome code containing an even number of bits generates. 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. Therefore, some combinations of multiple-bit failures result in a legitimate single-error 5-bit or 7-bit syndrome code. This results 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.

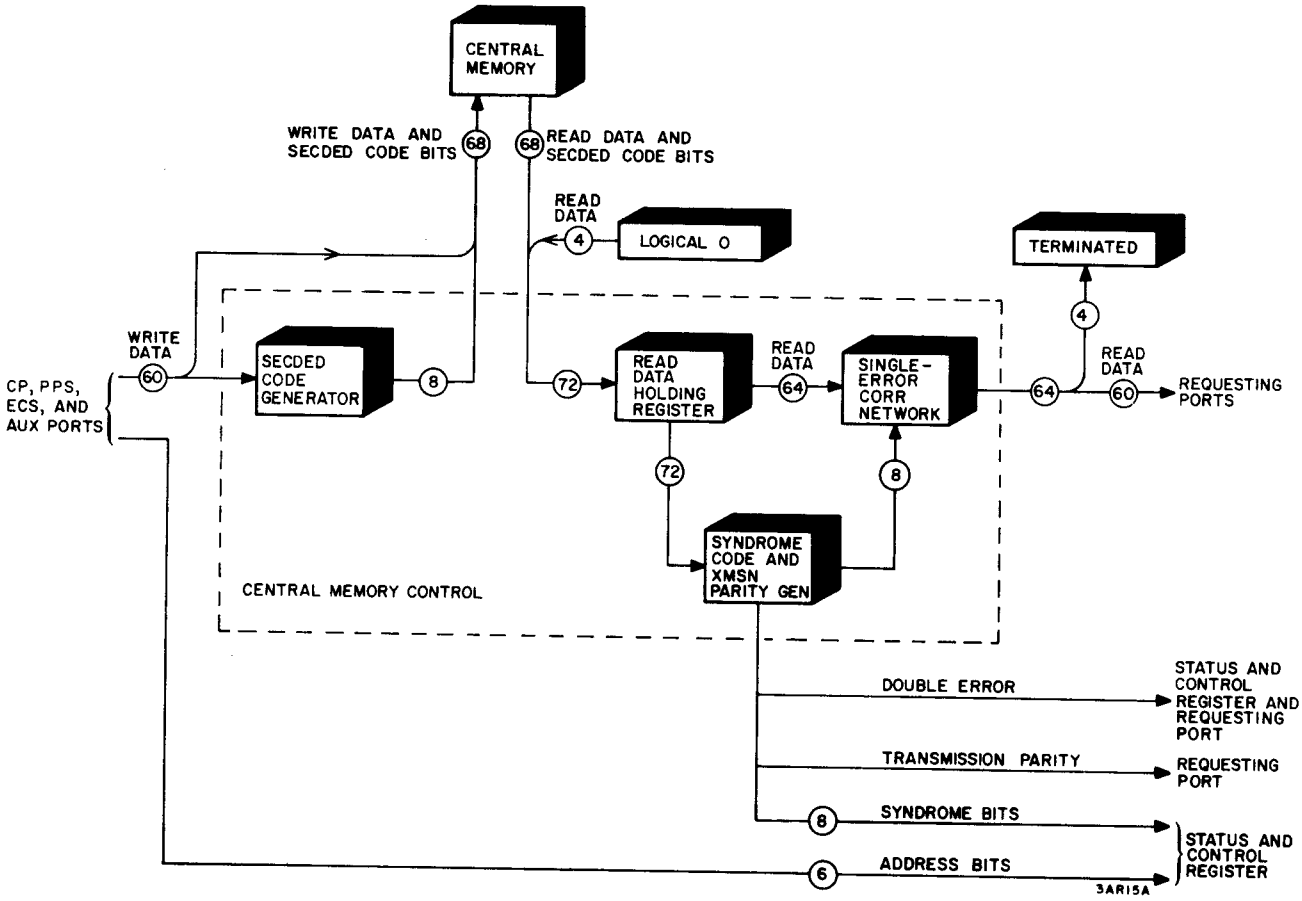


Figure 2-3. SECDED Network Block Diagram (SECDED Mode)

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	65 ①	100	66 ①	140	②	200	67 ①	240	②	300	②	340	50
001	60 ①	041	②	101	②	141	53	201	②	241	57	301	58	341	②
002	61 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	62 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	③	346	③
007	10	047	②	107	②	147	③	207	②	247	44	307	③	347	②
010	63 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	64 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

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 detected.

When there is no bit failure, the syndrome code equals zero and 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 eight 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, failing bit and failing chip on the module (in the case of single correctable errors). This information makes it possible for maintenance personnel to isolate the failure to a module level.

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 2-4 for all CMC error communications.

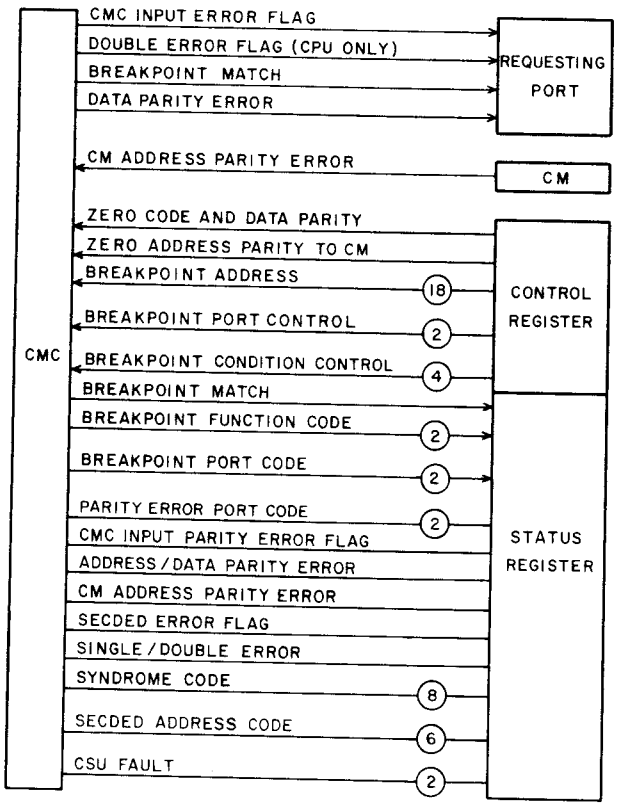


Figure 2-4. CMC Error Communications

ADDRESS PARITY

The CMC checks parity on the address paths from:

- CP-0, CP-1, PPS-0, and PPS-1 (models 171, 172, and 174)
- CP, PPS-0, and PPS-1 (model 173)
- 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
- Requesting port code
- Address 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 address parity error
- CSU-1 address parity error

If the CP is the requesting port to CM, a CMC error signal sets the parity error condition. If the exit mode bit 59 sets, additional action is taken in the CP. Refer to Exit Mode/Error Response under Central Processor in section 5 for further information.

DATA PARITY

The CMC checks parity on the data paths from:

- PPS-0 to CMC
- PPS-1 (if present) to CMC
- ECS (if present) to CMC
- CP-0 and CP-1 (if present) to CMC

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:

- CMC input 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 parity mode on a write operation, the data parity in models 171 through 175 generates in the CMC for transmission to CM and substitutes in place of SECDED code bit 0.

In parity mode on a read operation, the data parity bit in models 171 through 174 propagates (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 CPU if the CPU is the requesting unit. For other ports in model 175, the parity bit propagates 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 then generate for transmission to CM.

In SECDED mode on a read operation, all models send data through the SECDED network. A parity bit then generates in CMC and transmits 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 MEMORY - MODELS 171 THROUGH 175

CENTRAL MEMORY - MODELS 171 THROUGH 175

Models 171 through 175 have 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 are capable of storing 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.

TABLE 2-3. MODELS 171 THROUGH 174 CENTRAL MEMORY SIZES

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
65,536	8,192	Quadrant 0 Quadrant 1	
98,304	12,288	Quadrant 0 Quadrant 1 Quadrant 2	
131,072	16,384	Quadrant 0 Quadrant 1 Quadrant 2 Quadrant 3	
196,608	16,384 in CSU-0 8,192 in CSU-1	Quadrant 0 Quadrant 1 Quadrant 2 Quadrant 3	
262,144	16,384	Quadrant 0 Quadrant 1 Quadrant 2 Quadrant 3	

TABLE 2-4. MODEL 175 CENTRAL MEMORY SIZES

CM Size (Words)	Words Per Bank	CSU-0 / CSU-1 Memory Banks (Octal) 0 1 2 3 4 5 6 7 10 11 12 13 14 15 16 17
65,536	4,096	Quadrant 0
98,304	6,144	Quadrant 0 Quadrant 1
131,072	8,192	Quadrant 0 Quadrant 1
196,608	12,288	Quadrant 0 Quadrant 1 Quadrant 2
262,144	16,384	Quadrant 0 Quadrant 1 Quadrant 2 Quadrant 3

A CM consists of central storage unit-0 (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 171 through 174. In model 175, which always contains CSU-0 and CSU-1, the banks are numbered 0 through 17 (octal). The number of words that each bank is capable of storing 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 171 increase from 8,192 to 16,384 with the addition of quadrants 2 and 3. A special application of quadrant 1 in model 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 400-nanosecond CM cycle time permit up to eight CM banks to be active at any one time. Each CM bank has a 400-nanosecond access time at the CSU chassis ports and a maximum data transfer rate of one word each 50 nanoseconds.

DATA FORMAT

CM is capable of reading and writing 68 bits of information at each address. The 68 bits include 60 data bits and 8 SECDED code bits. The SECDED code bits are added before the 68 bits enter storage and are checked after the bits leave storage by the CMC. Figure 2-5 shows the data format.

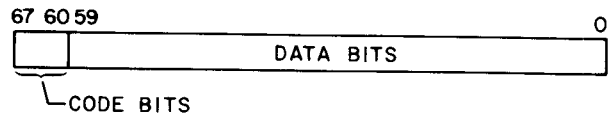


Figure 2-5. Models 171 through 175 CM Data Format

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 171 through 174 are shown in figure 2-6, and for model 175 in figure 2-7. 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 171 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.

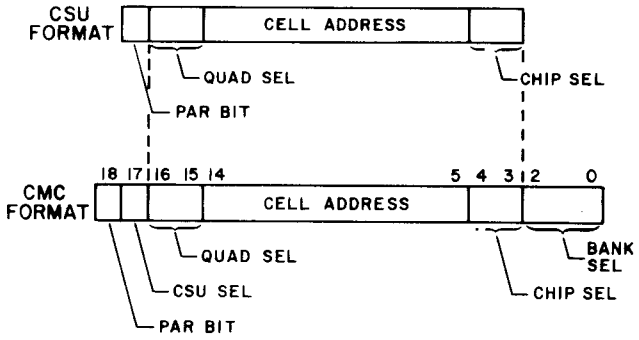


Figure 2-6. Models 171 through 174 CM Address Formats

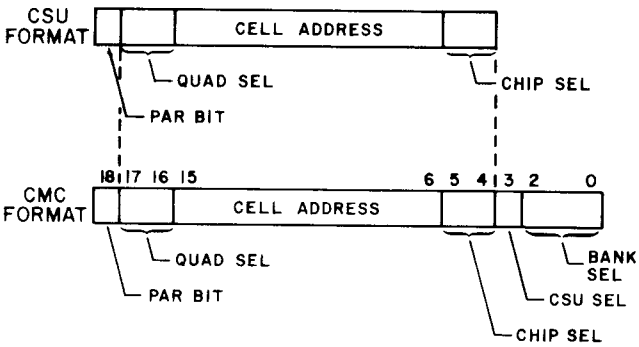


Figure 2-7. Model 175 CM Address Formats

ADDRESS PARITY

CM accepts the 14-bit address from CMC with one parity bit. Address parity is checked and an error signal is sent to CMC if a parity error is detected. If an 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.

REFERENCE OPERATIONS

Major CM reference operations which are under CMC control are read, write, and (if applicable) refresh. Refresh is a characteristic which applies only to model 171B and models 172 A and B through 175 A and B.

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.

A CM refresh is, in effect, a CM read operation without the transfer of data. The CM requires refreshing because its metal-oxide semiconductors (MOS) hold their capacitive charges for relatively short periods of time. Data would be lost if the charges were not restored. A refresh fully restores the charges.

Each CM refresh begins with the output of an interval counter which provides a refresh request each 25.6 microseconds. The request blocks all memory requests and starts a 400-nanosecond delay count which permits completion of memory requests in progress. If an exchange jump is in progress at the time of a refresh request, the request waits for the exchange jump to clear before initiating the delay count. At the completion of the delay count, the CMC sends a go refresh signal to each bank. The go refresh signal initiates a 400-nanosecond refresh cycle. The delay count and refresh cycle times block all memory requests for a total of 800 nanoseconds.

A seven-bit refresh address counter controls the CM refreshing. The counter provides the lower five bits of the chip addresses and the two chip select bits to permit refreshing of one-fourth of the CM chips every 32 refresh cycles. Refreshing of the entire CM therefore requires 128 (4 times 32) refresh cycles. The 128 refresh cycles occur once each 3.2768 milliseconds.

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 useable 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 are:

Models 171 through 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 section 3.

REFRESH FAULT

A refresh fault may occur only in model 171B and models 172 A and B through 175 A and B. 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 in the set condition, CM cannot be refreshed; therefore, data is no longer valid and must be reloaded after the fault condition is cleared.

CENTRAL MEMORY - MODEL 176

CENTRAL MEMORY - MODEL 176

Model 176 has basic and optional CM sizes. The CM size is determined by the number of 68-bit words, 60 data bits and 8 SECDED bits, that the CM is capable of storing. Table 2-5 lists the memory sizes. The basic CM size is the smallest size in the table. The optional sizes are the next three larger sizes. The basic CM size of 131,072 words is contained in 16 banks of one chassis. In addition, one 16-bank chassis is required to increment CM to each larger size.

TABLE 2-5. MODEL 176 CENTRAL MEMORY SIZES

CM Size (Words)	Words Per Bank	Memory Banks (Octal) 0 1 2 3 4 5 6 7 10 11 12 13 14 15 16 17
131,072	8,192	Chassis 9 Chassis 10
196,608	12,288	Chassis 9 Chassis 10 Chassis 11
262,144	16,384	Chassis 9 Chassis 10 Chassis 11 Chassis 12

The memory banks independently perform a read or write operation without affecting operations in the other banks. This permits memory references to occur concurrently in different banks. A one-word read operation within a bank requires an 82.5-nanosecond CM cycle time. A one-word write operation requires a 165-nanosecond CM cycle time.

The CM has bank phasing which assigns sequential addresses to different banks. For example, address 00000 is in the first bank, address 00001 is in the second bank, address 00002 is in the third bank, and so on through all banks. The address sequence then picks up again in the first bank and again progresses through all banks. Because the banks are independent, a bank can begin a memory cycle before adjacent banks have completed previously initiated cycles. Bank phasing thus allows references to sequential addresses to be heavily time-overlapped. Sequentially addressed data can transfer data at a rate of one word each 27.5 nanoseconds. During random addressing, some memory references are delayed because a previous reference to the same bank is not complete, causing a reduction in the transfer rate.

DATA FORMAT

The data format is the same as for the other models as shown in figure 2-5. The format includes 68 bits of information at each CM address. The 68 bits include 60 data bits and 8 SECDED code bits. A control section within CM adds the code bits before they are stored and checks them after they leave storage.

ADDRESS FORMAT

The CM address in model 176 originates in the CP. The address format (figure 2-8) is similar to that of model 175 but without a parity bit. The data address bits perform the same functions of selecting chip columns, chips, and specific bits as the model 175 address format.

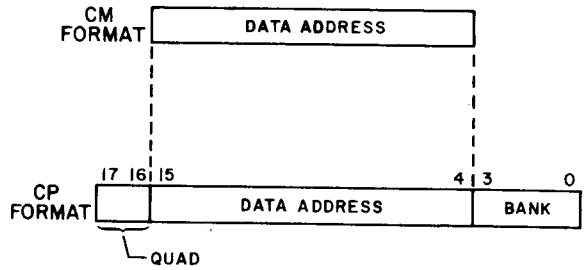


Figure 2-8. Model 176 CM Address Format

SECDED MODE

The SECDED mode of operation in CM provides single-error correction and double-error detection of memory errors.

Single-error correction corrects single-bit failures in words read from CM. Error correction occurs automatically and in no way degrades or otherwise affects system operation. Under normal operating conditions, single-bit errors are reported to the status and control register.

Double-error detection detects the failure of two bits in words read from CM but does not correct such errors. Double errors are reported in the following manner if bit 138 in the status and control register is clear.

- The CM parity error flag, bit 10 in the PSD register, sets.
- An exchange jump to the error exit address (EEA) register occurs immediately after execution of the current instruction word completes.
- The failing address is captured by the status and control register.
- A group of eight SECDED syndrome bits is captured in a syndrome register. This register is reported to the status and control register.

The CM normally operates in SECDED mode with logging of single-bit errors. However, four additional modes are available through status and control register program control: parity mode, maintenance mode, test mode, and inhibit log single-bit error (SBE) mode.

PARITY MODE

Parity mode is selected under program control through the status and control register. The semiconductor memory operates in an 8-bit parity mode. Parity errors are detected only on read memory references including all exchange jump references. The address for each read memory reference enters the error address register. If a parity error occurs, the parity bit(s) for the failing byte(s) is locked into the parity error/syndrome bit register, and the failing memory address is locked into the parity address register. The contents of these registers are accessed by the status and control register and are held until that register sends a clear parity signal. Occurrence of a parity error also causes bit 2^{10} of the PSD register to set, which causes an exchange jump to the error exit address. Parity errors that occur while the parity error/syndrome bit register is locked up from a previous error are not detected.

When the parity mode signal from the status and control register is absent, the memory operates in SECDED mode. The access time is identical in either mode.

MAINTENANCE MODE

Maintenance mode is selected under program control through the status and control register. Single-bit errors are reported the same as double-bit errors. Bit 2^{10} of the PSD register sets, and the CPU performs an exchange jump to EEA. The error address and syndrome bits are reported to the status and control register.

TEST MODE

Test mode is selected under program control through the status and control register. When the memory is operating in SECDED mode, the test mode signal forces all eight error correction code bits to logical zeros prior to writing into memory. Error correction is still performed in this mode of operation.

If memory is operating in eight-bit parity mode, the test mode signal forces all eight parity bits to be complemented prior to writing into memory. This feature allows the diagnostic program to force errors in order to check the SECDED hardware.

INHIBIT LOG SBE MODE

Single-bit errors are normally reported to the status and control register. When bit 118 of the status and control register sets, the inhibit log SBE mode prevents single-bit errors from being reported.

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 useable CM. CM reconfiguration is accomplished by setting switches to manipulate upper address bits. Hardware configures the CM quadrants so that sequential addressing is maintained. Model 176 reconfiguration options are 262K to 196K to 131K.

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 section 3.

LARGE CORE MEMORY EXTENSION — MODEL 176

LARGE CORE MEMORY EXTENSION — MODEL 176

LCME is a two-, four-, or eight-bank linear-select memory with a capacity of 524,288, 1,048,576, or 2,097,152 72-bit words. Each word contains 60 data bits, 8 error correction bits, 2 complement control bits, and 2 unused bits. Each bank is independent of the other banks. A storage reference to a bank results in a read/write cycle that requires 64 clock periods to complete. Sixteen 72-bit words are simultaneously read from or written into a memory bank. These words are held in a 1152-bit bank operand register. A subsequent reference to one of these words can be made without the delay of another read/write cycle. Maximum data transfer rate is one word each clock period. This maximum rate occurs during block copies between CM and LCME.

ADDRESS FORMAT

The location of each word in LCME has a 21-bit address. The address format (figure 2-9) depends on the memory size (and for the 512K memory, whether it contains the actual or error address). Within the address format, the lowest four bits specify one of sixteen 72-bit words within an LCME word. Bits 4, 5, and 20 specify one of two, four, or eight banks. The 14-bit bank address (bits 5 through 18 or 6 through 19) specify the location within the bank. For numerically consecutive addresses, consecutive banks are referenced every fourth address for systems using four or eight banks.

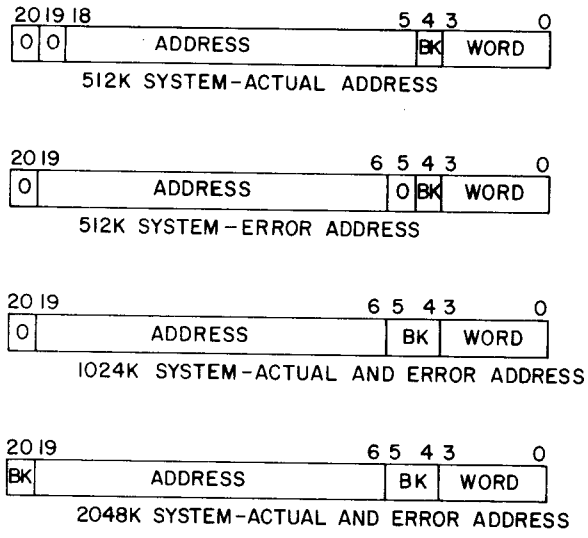


Figure 2-9. Model 176 LCME Address Format

SECDED MODE

The SECDED mode of operation in LCME provides single-error correction and double-error detection against memory errors.

Single-error correction corrects single-bit failures in words read from LCME. Error correction occurs automatically and in no way degrades or otherwise affects system operation. Under normal operating conditions, no status indications report the occurrence of single errors. However, for diagnostic and maintenance purposes, the status and control register can specify that single errors be reported.

Double-error detection detects the failure of two or more bits in words read from LCME but does not correct such errors. Double errors are reported in the following manner.

- The LCME error flag, bit 11 in the PSD register, sets.
- An exchange jump to the EEA register occurs immediately after execution of the current instruction word completes.
- The failing address is captured in the EEA register. The contents of this register is sampled by the status and control register.
- A group of eight SECDED syndrome bits is captured in a syndrome register. This register is reported to the status and control register. Under normal operating conditions, the only value of these bits is to indicate to the status and control register the occurrence of a SECDED error. A nonzero quantity in the syndrome bit holding register signals a SECDED error.

The LCME normally operates in SECDED mode with logging of single-bit errors. However, five additional modes are available through status and control register program control: parity mode, maintenance mode, test mode, inhibit log single-bit error (SBE) mode, and test complement mode.

PARITY MODE

When parity mode is selected, parity is checked each time a 60-bit word is read from LCME. When an LCME parity error is detected, the LCME parity condition flag, bit 11 in the PSD register, sets. The error address enters and locks in the error address register, and the parity bit(s) enter the parity/error syndrome bit register. The contents of these registers are accessed by the status and control register and are held until the status and control register sends the clear parity error signal. Occurrence of a parity error causes an exchange jump to EEA. Parity errors that occur while the

parity error/syndrome bit register is locked up from a previous error are not detected. When the parity mode signal from the status and control register is absent, the LCME operates in SECDED mode. Access time is identical in either parity or SECDED mode.

MAINTENANCE MODE

When maintenance mode is selected, single-bit errors are reported in the same manner as double-bit errors in SECDED mode. Bit 11 of the PSD register sets, and the CPU performs an exchange jump to EEA. The error address and syndrome bits are reported to the status and control register.

TEST MODE

When test mode is selected, the diagnostic program forces errors in order to check the SECDED hardware. When the LCME is operating in SECDED mode, the test mode signal forces all eight error correction code bits to logical zeros prior to writing into memory. Error correction is still performed in this mode of operation.

If memory is operating in parity mode, the test mode signal forces the complement of all eight parity bits prior to writing into memory.

INHIBIT LOG SBE MODE

Single-bit errors are normally reported to the status and control register in SECDED mode. When bit 178 of the status and control register sets, the inhibit log SBE mode prevents single-bit errors from being reported.

TEST COMPLEMENT MODE

When test complement mode is selected, the recomplementing of data read from LCME is inhibited. The data is transmitted to the X register or to CM in the same form as it appears in the LCME bank. This allows diagnostic software to check the operation of the population count performed on the write data and data paths for the upper and lower 36 bits of the LCME words.

BLOCK COPIES

Block copy instructions move quantities of data between LCME and CM at high speeds. All other activity in the CPU, except for I/O word requests, stops during a block copy operation. All instructions issued prior to this instruction execute to completion and no further instructions issue until the block copy is nearly complete. Also, an exchange interrupt request will not be honored until the block copy is nearly complete.

In systems with 1048K words of LCME, data flow between LCME and CM can occur at the rate of one 60-bit word each clock period. Systems with 512K words of LCME have a rate of approximately 32 words each 64 clock periods.

DIRECT (SINGLE-WORD) TRANSFERS

A read LCME instruction for a word not currently residing in a bank operand register requires 23 clock periods to deliver a 60-bit word to the designated X register. A read instruction for a word already residing in a bank operand register as a result of a previous instruction requires 6 clock periods (up to 15 clock periods if lockout occurs) to deliver the requested word to the designated X register.

The execution time for writing a word in LCME from an X register normally requires 3 clock periods. A delay of up to 37 additional clock periods is possible if a lockout condition occurs. A delay occurs if the required LCME bank is busy completing a bank read/write cycle for a different block of words than that required for the current instruction. In this case, the word is held in the LCME write register until the LCME bank is free.

BANK SELECTION

LCME bank selection may be specified manually or by program control to configure or degrade the memory. Manual selection requires the setting of LCME BANK SELECT switches, described in section 3. Program control requires the setting of status and control register bits 88 through 90, described in section 5.

INPUT/OUTPUT MULTIPLEXER — MODEL 176

INPUT/OUTPUT MULTIPLEXER — MODEL 176

The input/output (I/O) multiplexer (MUX) permits the PPUs and PPs to communicate with CM. The MUX controls the communication to the PPUs on four to fourteen 12-bit bidirectional channels (designated 2 through 17, octal), to the PPs on one or two 60-bit bidirectional ports (designated 0 and 1), and to CM through a single 60-bit bidirectional access. The number of channels and ports depends upon the PPU and PP options installed.

The basic I/O MUX channel configuration includes one 60-bit port and four high-speed 12-bit channels. Equipment options permit the addition of one 60-bit port, eight high-speed channels, and two normal-speed channels. High-speed channels transfer data approximately twice as fast as normal-speed channels.

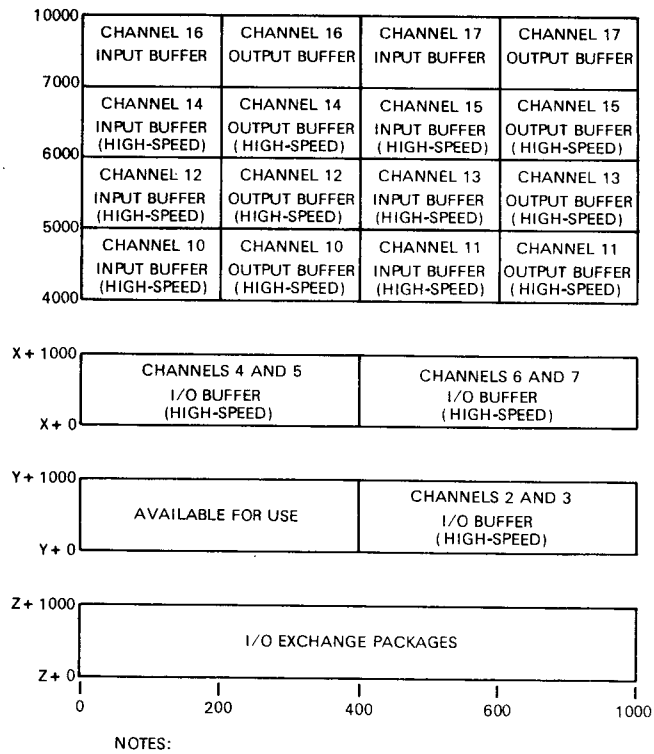
During communications between the PPUs and CM, the I/O MUX disassembles 60-bit transmissions from CM to 12-bit bytes. The MUX transmits the bytes through the channels to the PPUs. On transmissions from the PPUs to CM, the I/O MUX assembles the 12-bit bytes from the channels into 60-bit words before transmitting them to CM. The PP and CM communications occur through the 60-bit ports and do not require the assembly or disassembly of data.

Priority for CM access and I/O interrupts is port 0, followed by port 1 and the channels, with the lowest-numbered channels having the highest priority. Inputs have priority over outputs.

Each I/O MUX channel for a PPU has a buffer area reserved in the lowest 20,000 (octal) words of CM (figure 2-10). The locations of the buffer areas for channels 2 through 7 are determined by channel bias bits from the status and control register. The locations of the buffer areas for channels 10 (octal) through 17 (octal) are fixed.

The buffer areas each have two fields, lower and upper. Data enters or exits the buffer areas in a circular mode. This means that the first word in the upper field follows the last word in the lower field, and the first word in the lower field follows the last word in the upper field. Whenever a PPU fills or empties a buffer area and crosses a field boundary, a CPU interrupt occurs and an exchange sequence initiates a program to process the buffer data. The PPU continues to fill or empty the second buffer field while data in the first buffer field processes.

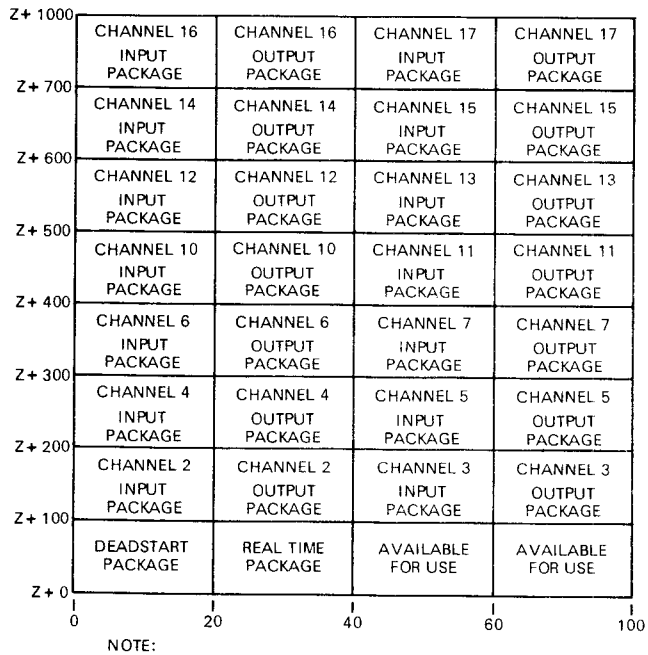
A separate exchange package for the I/O program exists for each I/O channel. The I/O exchange packages are permanently assigned in the lower-order addresses of reserved buffer area in CM. The I/O exchange packages are arranged as shown in figure 2-11.



NOTES:

1. ALL ADDRESS AND CHANNEL NUMBERS ARE OCTAL.
2. X, Y, AND Z ARE EQUAL TO 0 OR SOME MULTIPLE OF 1000 (OCTAL) THROUGH 17000 (OCTAL).

Figure 2-10. Model 176 CM I/O Buffer Addresses



NOTE:
ALL ADDRESS AND CHANNEL NUMBERS ARE OCTAL.

Figure 2-11. Model 176 CM I/O Exchange Package Areas

NORMAL PPU TO CM DATA TRANSFER

The following description lists the events in a normal PPU to CM input record sequence. The sequence begins with a reset input channel buffer instruction that resets the input channel buffer for receipt of a new record. This sets the input assembly counter and the input buffer address to zero. The CPU then notifies the PPU that CM is ready to receive data. It does this by transmitting a message to the PPU over the associated MUX output channel. The contents and format of the message depend upon the communication scheme, which is determined by the software.

Upon receipt of the message, the PPU enters the first 12-bit word into its output register. This entry causes the transmission of a word pulse and 12 data bits to the input channel control for this channel in the MUX. The MUX enters the first word in the upper 12 bits of the 60-bit assembly register. The MUX then sends a resume pulse to the PPU and advances the assembly counter. The resume pulse clears the output word flag at the PPU, and the second 12-bit word enters the PPU output register. The sequence of word pulse, input assembly, and resume pulse is repeated for each 12-bit word transmitted over the data path. When five 12-bit words have been assembled into a 60-bit word, a resume pulse is sent to the PPU and a word request is made for CM access. The MUX does not accept the next 12-bit word from the PPU until the request for CM access has been accepted by the CM. This may be only a few clock periods, or many clock periods, depending upon CM bank conflicts. Once the word request has been accepted by CM, the buffer address is advanced, the assembly counter is reset to zero, and the transmit and assembly procedure is repeated for the next 60-bit word.

When the PPU transmits enough words to fill half of its assigned CM buffer area, the MUX sends an interrupt request to the CPU. When this is accepted by the CPU, an exchange jump is initiated to a program that processes the data in the first half of the buffer. Meanwhile, the PPU continues to transmit 12-bit words, which the MUX assembles into 60-bit words and stores in the upper half of the buffer. When the upper half of the buffer becomes full, the MUX sends another interrupt request to the CPU, provided that the program from the first interrupt has completed processing the lower half of the buffer and has performed an exchange exit. Otherwise, the interrupt request is not sent to the CPU, and further input from the PPU is locked out until the exchange exit is executed.

NOTE

If an error condition occurs which causes the I/O program to exit to an error handling routine at EEA, the error routine may, in returning to the I/O program, inadvertently release the interrupt lockout condition prematurely by performing an exchange exit (013) instruction. To prevent this situation from occurring, bit 17 of EEA can be set in the incoming exchange package. This bit is sent to exchange jump control and blocks an 013 instruction from releasing any I/O interrupt request flags that might be set.

When the interrupt request has been sent, the PPU begins to enter data into the lower half of the buffer while the data in the upper half is being processed. Thus, the buffer operates in a circular mode with interrupts at the center and end of the buffer area.

An input record may contain any amount of data. The transmitting PPU terminates the record by sending a record pulse to the MUX. Before sending the record pulse, the PPU ensures that the last 12-bit word was accepted by the MUX. (If the PPU output word flag is clear, the MUX has accepted the last word.) Upon receipt of the record pulse, the MUX sends an interrupt request to the CPU. If the PPU has not transmitted enough 12-bit words to form a complete 60-bit word, the remainder of the word is filled with zeros. Other than this, the CPU handles this request the same as an interrupt request caused by a threshold condition. The resulting I/O program determines whether the interrupt was caused by a buffer threshold or a record pulse. It does this by reading the CM address (read input channel status instruction) to determine whether a threshold has been crossed since the last interrupt. The I/O program processes the input data according to the situation sensed.

The PPU must not begin transmitting a new record of input data until the data in the buffer has been processed. There is no hardware provision to prevent the PPU from doing this. Therefore, the PPU program must not enter new data until directed to do so by the CPU program. If the PPU proceeds before the CPU has reset the input buffer, the incoming data for the new record may be partially lost. The incoming record continues to be input with no indication of error except that the record is shortened by the lost data.

NORMAL CM TO PPU DATA TRANSFER

The following description lists the events in a normal CM to PPU output record sequence. The I/O program has already loaded the output buffer with some data. The output sequence begins with a reset output buffer instruction that sets the output buffer address to zero and sends a word request to CM to read the first word from the output buffer to the 60-bit disassembly register in the MUX. When CM delivers the 60-bit word to the disassembly register, the output channel control for this channel clears the disassembly counter and outputs a record pulse and a word pulse to the PPU to indicate that transmission of a new record is starting.

The upper 12 bits of the data in the disassembly register are placed on the input channel for the PPU. When the PPU program senses the record pulse on its input channel, it reads the 12 bits of data and sends a resume pulse to the MUX. The MUX output data remains on the PPU input channel until the PPU accepts it.

When the resume pulse arrives from the PPU, the MUX advances the disassembly register to the next 12 bits of the 60-bit word and sends another word pulse to the PPU. The output buffer address also advances to the next address at this time so that a program monitoring this channel could determine that the PPU has accepted the first 12 bits of a new 60-bit word. The sequence of output disassembly, word pulse, PPU input, and resume pulse continues until the entire 60-bit word has been sent by the

MUX. At this time, the MUX sends another word request to CM for the next word in the output buffer. When this word arrives in the disassembly register, the upper 12 bits and a word pulse are sent to the PPU, and the process of delivering a new 60-bit word is repeated.

When the PPU has emptied half of its assigned buffer area, the MUX sends an interrupt request to the CPU. When this is accepted by the CPU, an exchange jump initiated to the program that refills the portion of the buffer that has just been emptied. This operation is similar to that performed for a PPU to CM transfer. Output to the PPU continues from the upper half of the buffer while the lower half is being refilled.

When the upper half of the buffer becomes empty, the MUX sends another interrupt request to the CPU provided that the program from the first interrupt has completed processing the lower half of the buffer and has performed an exchange exit. Otherwise, the interrupt request is not sent to the CPU, and further output to the PPU is locked out until the exchange exit is executed.

NOTE

If an error condition occurs which causes the I/O program to exit to an error handling routine at EEA, the error routine may, in returning to the I/O program, inadvertently release the interrupt lockout condition prematurely by performing an exchange exit (013) instruction. To prevent this situation from occurring, bit 17 of EEA can be set on the incoming exchange package. This bit is sent to exchange jump control and blocks an 013 instruction from releasing an I/O interrupt request flags that might be set.

Using a software-determined communication scheme, the CPU has notified the PPU of the length of the record. When the PPU receives the expected amount of data, it stops reading data on its input channel, stopping further transmission by the MUX.

HIGH-SPEED PPU TO CM DATA TRANSFER

The following description lists the events in a high-speed input record sequence. The sequence for a high-speed channel is basically the same as for a normal channel except that the word and record pulses from the PPU are not synchronized by the MUX.

The sequence begins with a reset input channel instruction that resets the input channel buffer for receipt of a new record. This sets the input assembly counter to zero and the input buffer address to the starting address of the buffer for the selected channel.

Next, the PPU enters the first 12-bit word into its output register. This causes the transmission of a word pulse and 12 data bits to the input channel control for this channel in the MUX. The MUX enters the 12-bit word in the upper 12 bits of the 60-bit assembly register.

A static high-speed resume signal is sent to the PPU during this time. This clears the output word flag in the PPU immediately after it sets. The second 12-bit word may now be entered in the PPU output register. This sequence continues as each 12-bit word transmits over the data path.

When five 12-bit words are assembled into a 60-bit word, the MUX sets the input word request flag for CM access. This blocks the high-speed resume signal to the PPU and clears the input assembly counter in preparation for the arrival of the next PPU word. It also blocks the processing of a new 12-bit word if one arrives before the request for access has been accepted by CM. This may be only a few clock periods or many clock periods, depending upon CM bank conflicts and channel priority. Once the word request is accepted by CM, the buffer address advances, the input word request flag clears, and the high-speed resume signal is again sent to the PPU. The transmit and assembly procedure then repeats for the next 60-bit word.

Interrupt requests resulting from reaching a buffer threshold or receiving a record pulse from the PPU are the same as for the normal PPU to CM data transfer.

HIGH-SPEED CM TO PPU DATA TRANSFER

The following description lists the events in a high-speed output record sequence. The sequence for a high-speed channel is basically the same as for a normal channel except that the resume pulse is not resynchronized by the MUX. Also, the output data path includes a series of three output data buffer registers. The output channel control also controls the flow of data from the disassembly register through these buffer registers to the PPU. The three buffer registers are designated ranks A, B, and C.

The output sequence begins with a reset output buffer instruction that sets the output buffer address to the starting address of the buffer. At this time, the MUX also sends a word request to CM to read the first word from the buffer to the 60-bit disassembly register. When the 60-bit word has been delivered to the disassembly register, the MUX clears the disassembly counter and sends a word pulse and a record pulse to the high-speed control. Concurrently, the upper 12-bit word in the disassembly register transmits to rank A of the buffer registers.

Upon receipt of the record pulse from the output channel control, the high-speed buffer control transmits a record pulse to the PPU. This sets the input record flag in the PPU.

Upon receipt of the word pulse from the output channel control, the high-speed buffer control enters the 12-bit word from the disassembly register into rank A. It then transmits the word pulse to the PPU where it sets the input word flag. The word pulse is not sent to the PPU if an interrupt lockout condition exists in the output channel control.

In consecutive clock periods, the data moves from rank A to rank B to rank C of the buffer registers. The data in rank C is transmitted to the PPU and remains in the data path until the PPU transmits a resume pulse to the high-speed control.

The high-speed control does not wait for the resume pulse from the PPU before sending a resume pulse to the output channel control. The output channel control increments the disassembly count and transmits the second 12-bit word to rank A. At this time, the output channel control advances the address register to the next address in the CM buffer and sends a word pulse to the high-speed control. Upon receipt of this second word pulse, rank A is entered with the second 12-bit word, and the resume pulse is again sent to the output channel control. In the following clock period, the data in rank A moves into rank B.

The process is repeated for the third 12-bit word. However, when the output channel control sends the third word to rank A, the resume pulse is not sent back to the output channel control.

At this point all action stops until the PPU accepts the first 12-bit word and transmits a resume pulse to the high-speed buffer control. When a resume pulse arrives from the PPU, rank C clears and is entered with the second 12-bit word in rank B. A resume pulse is then sent to the output channel control.

The sequence continues until the fifth 12-bit word has been sent to rank A by the output channel control. At this time, the output channel control sends another word request to CM for the next 60-bit word in the buffer.

At the time the output word request flag sets, the last two 12-bit words are in ranks B and C. The PPU accepts the fourth word and transmits a resume pulse to the high-speed buffer control. Rank C is then cleared and entered with the fifth word from rank B. When this data has been delivered to the PPU, action halts until the requested word is delivered to the disassembly register from CM.

Some clock periods later, the word is delivered to the disassembly register, and the process of delivering a new 60-bit word to the PPU begins.

Interrupt requests results from reaching a buffer threshold are the same as for the normal CM to PPU data transfer.

LOGIC SCANNER — MODEL 176

LOGIC SCANNER – MODEL 176

The logic scanner is a static switching network with fan-in and fan-out capabilities. The switching network permits any one of the PPs to communicate with any one of the PPUs through a single, bidirectional, 12-bit I/O channel.

Selection of one of 12 channels that connects PPU to the PPS is determined by four control bits. These bits are sent to the logic scanner from the status and control register.

Signals between the logic scanner and the PPUs are asynchronous.

DATA CHANNEL CONVERTER —MODELS 171 THROUGH 176

DATA CHANNEL CONVERTER — MODELS 171 THROUGH 176

Each system DCC attaches to a data channel of the PPS (figure 2-12). A DCC may share the data channel with up to seven other pieces of CDC 6000/CYBER series peripheral equipment. As many as eight 3000 series controllers can be attached to one DCC.

To prepare any of the 3000 series equipment for operation, the DCC must first be selected. The desired 3000 series equipment is selected (connected). The two select operations are made by function codes sent from a PP 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 clears by any one of the following.

- A DCC MC function (1700). This clears all 3000 series equipment attached to the DCC and the DCC itself.

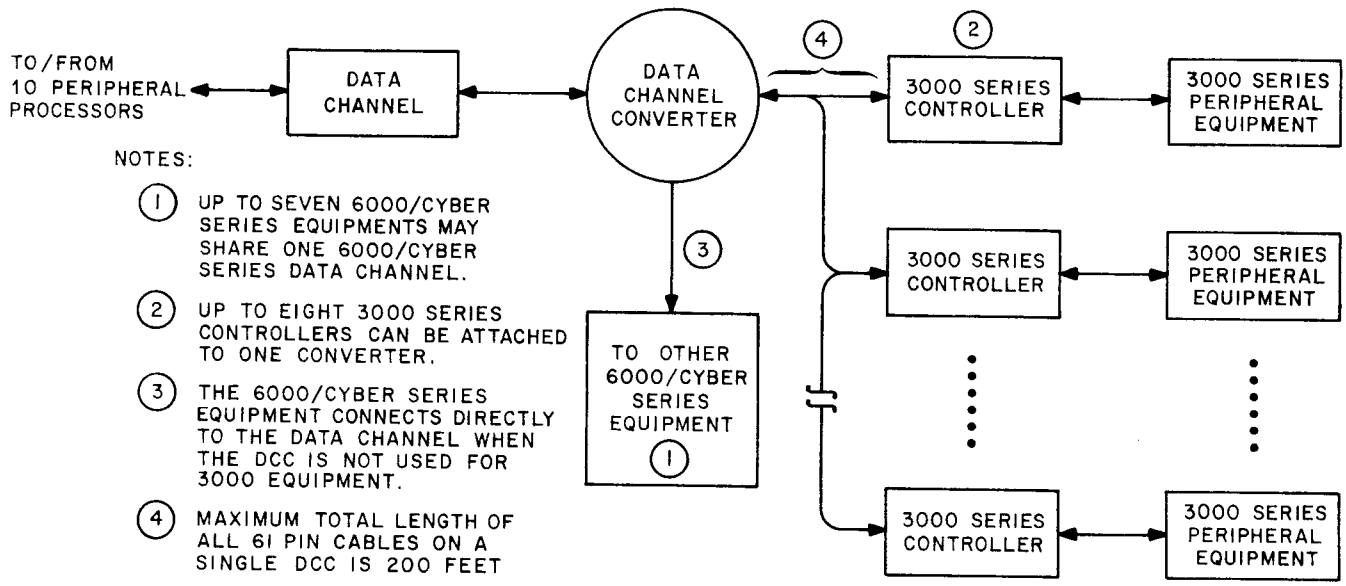


Figure 2-12. Data Channel Converter Configuration

- A function code sent to the interrupting equipment.
- A deadstart MC signal from the CDC 6000/ CYBER data channel.

3000 POWER FAILURE MODE

The power failure mode enhancement allows each DCC to check for a power failure on a connected piece of external equipment. The detection of this power failure sets main power fail bit 36 (44, octal) in the status and control register. The power failure also terminates I/O operations on the DCCs under certain conditions. Refer to Data Channel Programming in section 5 for further details.

NOTE

This enhancement does not apply to early model 172 through 175 systems.

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. Data transmitted during the buffer-flush operation is undefined.

DISPLAY CONTROLLER — MODELS 171 THROUGH 176

DISPLAY CONTROLLER — MODELS 171 THROUGH 176

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 lines are for the x (horizontal) deflection, and two lines 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.

PERIPHERAL PROCESSOR UNITS – MODEL 176

PERIPHERAL PROCESSOR UNITS – MODEL 176

The model 176 basic system contains four PPUs and has the option of adding individual PPUs up to a total of 10. Each PPU is a self-contained functionally independent computer with a memory. A PPU shares access paths to CM and the peripheral processor subsystem (PPS) with each installed PPU. Each PPU has its own hardware for arithmetic operations, logic operations, and I/O channels.

A primary function of the PPU is to perform I/O tasks at the request of the CP. The PPU may also directly control peripheral equipment with a minimum of intervening circuits and perform a modest amount of character conversion and data formatting before transmitting data to the CP. Another function may be to perform the synchronization required to interface an electro-mechanical device to the CP. This function generally requires dedication of the PPU to one or more specific units such as printers, card readers, tape units, or disk files.

A PPU communicates through its eight I/O channels. Each I/O channel is bidirectional and carries 12 data bits. One channel can be used for communications with another PPU, one for communication with the logic scanner, one for communication with the I/O MUX, and four for communication with 7000 peripheral equipment. One channel is not available for external use. Only one channel is active at a time. On a write operation, the MUX assembles data into 60-bit words for CM. On a read operation, the MUX disassembles the 60-bit CM word into 12-bit bytes for the PPU.

Channel instructions direct all activities with other PPUs, the logic scanner, I/O MUX and external equipment. These instructions select any equipment on any channel and transfer data to or from the selected equipment, PPU, or logic scanner.

COMPUTATION SECTION

The computation section performs the arithmetic operations associated with manipulating operands and with indirect addressing. The arithmetic operations involve seven registers: A, P, Q, X, Sk, fd, and k. The A register is the only one used directly by a programmer.

A Register

The 18-bit A register is the principal operand register and is used for the I/O, shift, and logical arithmetic instructions. In an arithmetic operation, the A register always holds one of the operands and always receives the arithmetic result. The content of A is treated as a signed operand. If bit 17 is set, the operand is negative.

Overflows are ignored although an end-around carry may show in the register at the end of an instruction execution. No sign extension is provided for 6-bit or 12-bit quantities entered in the low-order bits. However, the unused upper bits do clear to zero. Zero is represented by all zero bits.

The A register counts the length of the block for block input or output instructions. At the transmission of each word, the A register enters the new count.

The A register receives the input data word (12 bits) for the input to A instruction and holds the output data word (12 bits) for the output from A instruction.

P Register

The 12-bit P register holds the address of the current instruction. During the execution of the current instruction, the content of P advances by one or two to provide the address of the next instruction in the program for 12- or 24-bit instructions. If the current instruction is a jump, P receives the jump address.

Q Register

The 12-bit Q register has two major functions. It holds the address of an operand during instruction execution and holds the upper six bits of an 18-bit operand in the lower six bits of the register during operand arithmetic.

X Register

The 18-bit X register holds all data read from memory. The register also holds the lower 12 bits of the operand during the 18-bit arithmetic operations of the A register.

Sk Register

The 6-bit Sk register contains a shift count during execution of shift instructions. The lowest-order five bits contain the number of bit positions by which the A register is to be shifted. The highest-order bit determines whether the shift is left circular or right open-ended.

fd Register

The 12-bit fd register holds the current instruction word for translation. The upper six bits are the f designator, and the lower six bits are the d designator from the instruction.

k Register

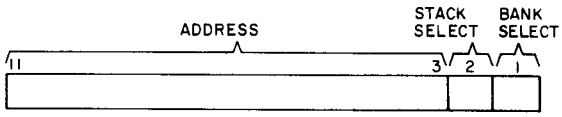
The 3-bit k register is the instruction cycle counter and is used to count the number of memory references required during execution.

PPU MEMORY

The PPU has its own 13-bit, 4096-word magnetic core, random access memory with a read/write

275-nanosecond cycle time. Each 12-bit data word has a parity bit attached.

The memory is organized into two banks. Each bank consists of two stacks of 1024 13-bit words each. Consecutive addresses alternate between the memory banks to increase processing speed. The memory address format is:



Each bank has an associated S register which holds the address of the operand in storage, a Z register which holds operands to be stored, and an X register which receives operands read from either bank. Therefore, there are two Z and two S registers for each PPU. Associated with each Z register is a parity-generating circuit that generates an odd parity bit that is stored in the memory with the operand. Parity is checked when operands are read from memory. In the event of a parity error, the PPU sends a parity error signal to the status and control register.

PPU INPUT/OUTPUT

A PPU communicates over bidirectional channels which connect to the I/O mux and other devices through I/O cables. Each PPU has provisions for eight input and eight output channels. Each cable provides 12 bits of incoming or outgoing data and the associated control lines for that data. The PPU may enter the data on any one of these eight input or output channels at any one time. Each path has two associated control lines carrying control information in the direction of data flow. These lines carry a word pulse to indicate passage of each 12-bit word of data and a record pulse to indicate the completion of a record of data. Each path has one associated control line carrying control information against the direction of the data flow. This line carries a resume pulse to indicate receipt of a data word.

Input Channel Control

The PPU may accept the data on any one of the eight input channels at any one time. Channel selection of the input channels, numbered 0 through 7, is determined by the lowest-order three bits in the d portion of the fd register.

Control of an input channel occurs by the setting and clearing of control flags within the PPU. The flags are directly associated with the control signals transmitted or received over the input channel. The control flags include the input word flag, input record flag, and input resume flag.

The input word flag sets when the PPU receives a word pulse on the input channel. The flag clears when the PPU accepts the data on the channel and sends a resume pulse to the transmitting device at the other end of the channel. A deadstart forces the flag to a cleared state. A PPU senses the status of the flag by executing I/O jump instruction 60 or 61.

The input record flag sets when the PPU receives a record pulse on the input channel. The flag clears when the PPU accepts the next input data word and sends a resume pulse to the data transmitter at the other end of the channel. A deadstart forces the flag to clear. A PPU senses the status of the flag by executing I/O jump instruction 62 or 63.

The input resume flag sets for 1 clock period when the PPU accepts the input data and is ready for the next word transmission. A deadstart sets the flag. The PPU transmits a resume pulse over the input channel during the time that the flag is set.

Output Channel Control

The PPU may enter data on any one of the eight output channels at any one time. Channel selection of the output channels, numbered 0 through 7, is determined by the lowest-order three bits in the d portion of the fd register. Data remains on the output channel until changed by the transmitting PPU.

Control of an output channel occurs by the setting and clearing of control flags within the PPU. The flags are directly associated with the control signals transmitted or received over the output channel. The control flags include the output word flag and output record flag.

The output word flag sets when the PPU transmits a 1-clock-period-wide word pulse over the associated output channel. The flag clears when the PPU receives a resume pulse over the output channel. A deadstart clears the flag. A PPU senses the status of the flag by executing I/O jump instruction 64 or 65.

The output record flag sets when the PPU transmits a 1-clock-period-wide record pulse over the associated output channel. The flag clears when the PPU receives a resume pulse over the output channel. A deadstart clears the flag. A PPU senses the status of this flag by executing I/O jump instruction 66 or 67.

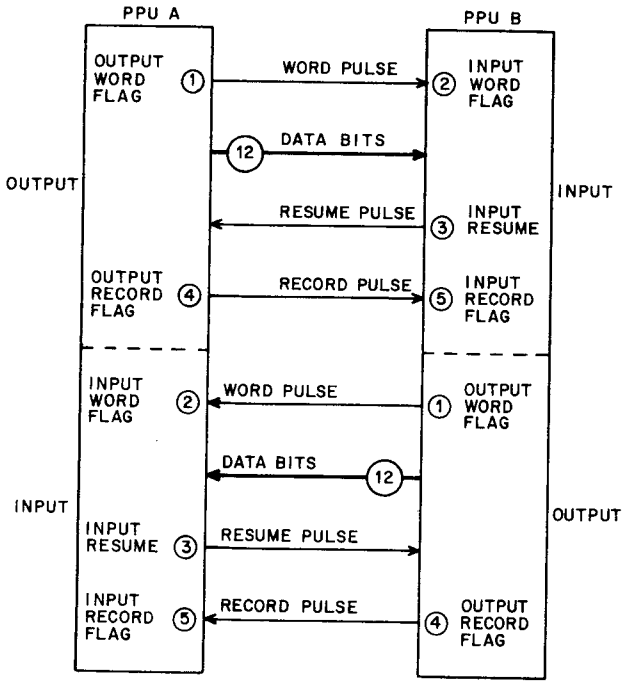
PPU TO PPU DATA TRANSFERS

Figure 2-13 shows two PPUs with an interconnecting channel. The channel condition is for a series of one-word transfers with A being the output PPU and B being the input PPU. The following sequence describes one method for a one-word data transfer between the two PPUs.

1. PPU A executes an output from A instruction (72). The instruction places 12 bits of data from the A register on the output channel, sets the output word flag, and sends a word pulse to PPU B.

2. PPU B is periodically executing a jump on input word flag instruction (60). Upon receipt of the word pulse from PPU A, the input word flag sets and PPU B jumps to an input program and executes an input to A instruction (70). This instruction enters the 12 bits on the input channel, clears the input word flag, and sends a resume pulse to PPU A.

3. The resume pulse clears the output word flag and the output record flag, if set at PPU A. After executing the output from A instruction (step 1), PPU A repeatedly executes a jump on no output word flag instruction (65). If PPU B has not yet accepted the output word, the output word flag remains set. If the output word flag clears, PPU A proceeds to the next instruction.



NOTES:

① SET BY ANY OUTPUT DATA INSTRUCTION (72, 73), CLEARED BY A RESUME PULSE

② SET BY A WORD PULSE, CLEARED BY RESUME PULSE

③ SET BY ANY INPUT DATA INSTRUCTION (70, 71), CLEARED AFTER ONE CLOCK PERIOD

④ SET BY OUTPUT RECORD FLAG INSTRUCTION (74), CLEARED BY A RESUME PULSE

⑤ SET BY OUTPUT RECORD PULSE, CLEARED BY RESUME PULSE

Figure 2-13. PPU/PPU Communications

In the figure, PPU A notifies PPU B of a word transmission with a word pulse. PPU A can also accomplish this by executing an output record flag instruction which sends a record pulse to PPU B. In this case, PPU B periodically monitors the status of the record flag instead of the word flag. Then, when the record flag sets upon receipt of the record pulse, PPU B goes to a data transfer sequence.

For block transfers, block input and block output hardware perform some of the flag monitoring functions automatically. The following sequence illustrates one method for a block transfer between two PPUs.

1. PPU A prepares for the block transfer by placing the length of the block to be transferred in the A register. The PPU then executes a block output instruction (73). This instruction sets the output word flag and sends 12 bits and a word pulse to PPU B.

2. Assuming that PPU B has been notified of the length of the block through a software-determined communication scheme, PPU B prepares for an input by placing the length of the expected block in its A register. PPU B then repeatedly executes a jump on input word flag instruction (60).

3. The word pulse from PPU A sets the input word flag at PPU B, and PPU B executes the block input instruction (71). This instruction enters the 12 bits, clears the input word flag and input record flag (if set), and sends a resume pulse to PPU A.

4. The resume pulse clears the output word flag and the output record flag (if set) at PPU A. The block output hardware automatically decrements the output count in the A register and sends the next 12 bits and another word pulse to PPU B.

5. Similarly, at PPU B, the block input hardware decrements the input count in the A register and enters the next 12 bits. The sequence repeats until the content of the A register of PPU A is zero, and PPU A sends a record pulse to PPU B.

If the two counts in the A registers are unequal, the PPU with the larger count hangs up, waiting for the proper response from the other PPU, which has already terminated its block transfer operation. Normally, however, if PPU A terminates first, it sends a record pulse to PPU B, which terminates input to PPU B. If PPU B terminates first, PPU A hangs up and remains hung up until PPU B inputs enough additional words to decrease the output count in PPU A to zero or until PPU A is deadstarted.

PPU TO PERIPHERAL EQUIPMENT DATA TRANSFERS

A direct-driven peripheral device requires two PPU channels. One channel performs control and status and the other data transfers. Depending upon the peripheral device, the associated control signals are terminated, set to 1 or 0, or assigned functions.

For detailed information on data transfers between a PPU and a peripheral device, refer to the documentation on the specific peripheral device.

PERIPHERAL PROCESSOR SUBSYSTEM — MODELS 171 THROUGH 176

PERIPHERAL PROCESSOR SUBSYSTEM — MODELS 171 THROUGH 176

The peripheral processor subsystem (PPS) consists of 10 peripheral processors (PPs). Each PP is a functionally independent computer that has its own memory. The PPs share access to CM and 12 bidirectional I/O channels. The PPs are organized into 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 all system models. 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 PP can access any I/O channel.

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 select any equipment on any channel and transfer data to or from the selected equipment.

Each PP exchanges data with CM through CMC in models 171 through 175 or through the I/O MUX in model 176 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 or the I/O MUX. 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/disassembly 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.

In models 171 through 175 only, data transmission parity is generated on all CM writes and is checked on all CM reads. If a data parity error is detected, a bit sets in the status and control register.

REAL-TIME CLOCK

The PPS 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 each 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 PP 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 do not execute and cause the instruction to hang.

DEADSTART

Deadstart is a PPS operation that provides initial starting of the computer, dumping of the contents of PPMs to an output device (normally a printer), or sweeping PPMs without executing instructions. Deadstart sequence is initiated by the DEAD START switch on the deadstart panel in bay 1 or the DEAD START switch on the display station. The panel includes controls for assigning any PPM to 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.)

PP 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 sets 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 assigned to PP-0. PP-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 with software so the PPS operates without the failing PPM. Degrading the PPS must be done through the operating system. A PPS reconfiguration and a PPS degradation permit computer operation to continue without the failing PPM. This permits correction of the failing PPM during scheduled maintenance.

In models A and B, the PPMs have MOS-type memory chips which require refreshing. A refresh cycle occurs once each 32 microseconds and requires a 500-nanosecond period. At a 1X operating speed, the refresh cycle is invisible to a 10-PP system. At a 1X operating speed in an expanded PPS (more than 10 PPs), the refresh cycle becomes apparent and can cause PP conflicts. These conflicts at a 2X operating speed are described in Channel Conflicts in an Expanded System in section 5. At a 2X operating 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 2-14). 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.

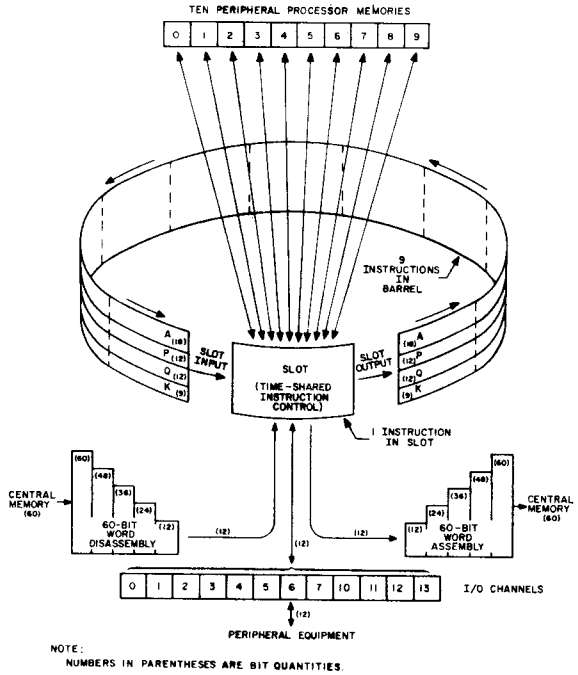


Figure 2-14. Barrel and Slot Operation

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 allows for PP program operating speeds of 1X and 2X. These speeds are program-selectable by a single bit in the status and control register. When clear, the bit causes the PPs to operate at 1X speed, which consists of a 50-nanosecond slot time for each PP once each 1000 nanoseconds. When set, the bit causes the PPs to operate at 2X speed, which consists of a 50-nanosecond slot time for each PP once each 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 (models A and B only).

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. PP MEMORY SELECT switches on the deadstart panel permit assigning any PPM to PP-0. Following a PPM selection, the 10 PPMs remain in order, assigned to consecutive PPs.

For example, if PPM-8 is assigned to PP-0, PPM-9 is assigned to PP-1 and PPM-0 is assigned to PP-2.

A Register

The 18-bit A register holds one operand for arithmetic, logic, or selected I/O operations. The content of A 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 memory control. At deadstart, the A register is set to 10000 (octal).

P Register

The 12-bit P register is the program address register, except during the execution of instructions 61, 63, 71, and 73. For these instructions, the P register contains the PPM address of the data transfer. At deadstart, the P register is set to zero.

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 instructions, shift count, and relative jump designator. At deadstart, each rank of the Q register is set to a corresponding PP number. Rank 0 is set to PP0, rank 2 is set to PP2, and so on.

K Register

The 9-bit K register holds a 6-bit f portion of an instruction word and a 3-bit trip count. The trip count determines the operation of an instruction at

different stages of completion. At deadstart, in load mode the K register is forced to 710; in sweep mode the K register is forced to 505; and in dump mode the K register is forced to 730.

PP 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 one parity bit at rates up to one word each 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 each 500 nanoseconds with two exceptions. One exception is on a 10-, 14-, 17-, or 20-PP system during a block transfer over a channel. The block transfer rate of 500 nanoseconds can be maintained for 63 words. A 500-nanosecond refresh cycle is then initiated for the PPMs. The other exception occurs only in a 14-, 17-, or 20-PP system and is described in Channel Conflicts in an Expanded PP System under Peripheral Processor Programming 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 manual deadstart or a force exit on the selected PP function through 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 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 controlling such things as breakpoint (available only in models 171 through 175), 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 system expanded to include PPS-1. This register is hardwired to channel 36 in the PPS-1 chassis. The PPS-1 register is smaller and contains only the bits that affect the PP in PPS-1. The test-error portion of both status and control registers may be interrogated with one test.

The status and control register bit usages differ between models 171 through 175 and model 176. Section 5 defines the status and control register bits and describes their use.

Some status bits and some control bits in the status and control register are displayed by modules with light-emitting diodes. The modules and the bits they display are described in section 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 groups control and indicator descriptions, power-on procedures, power-off procedures, and operating procedures by model number as follows:

- Controls and indicators - models 171 through 174
- Controls and indicators - model 175
- Controls and indicators - model 176
- Power-on and power-off procedures - models 171 through 176
- Operating procedures - models 171 through 176

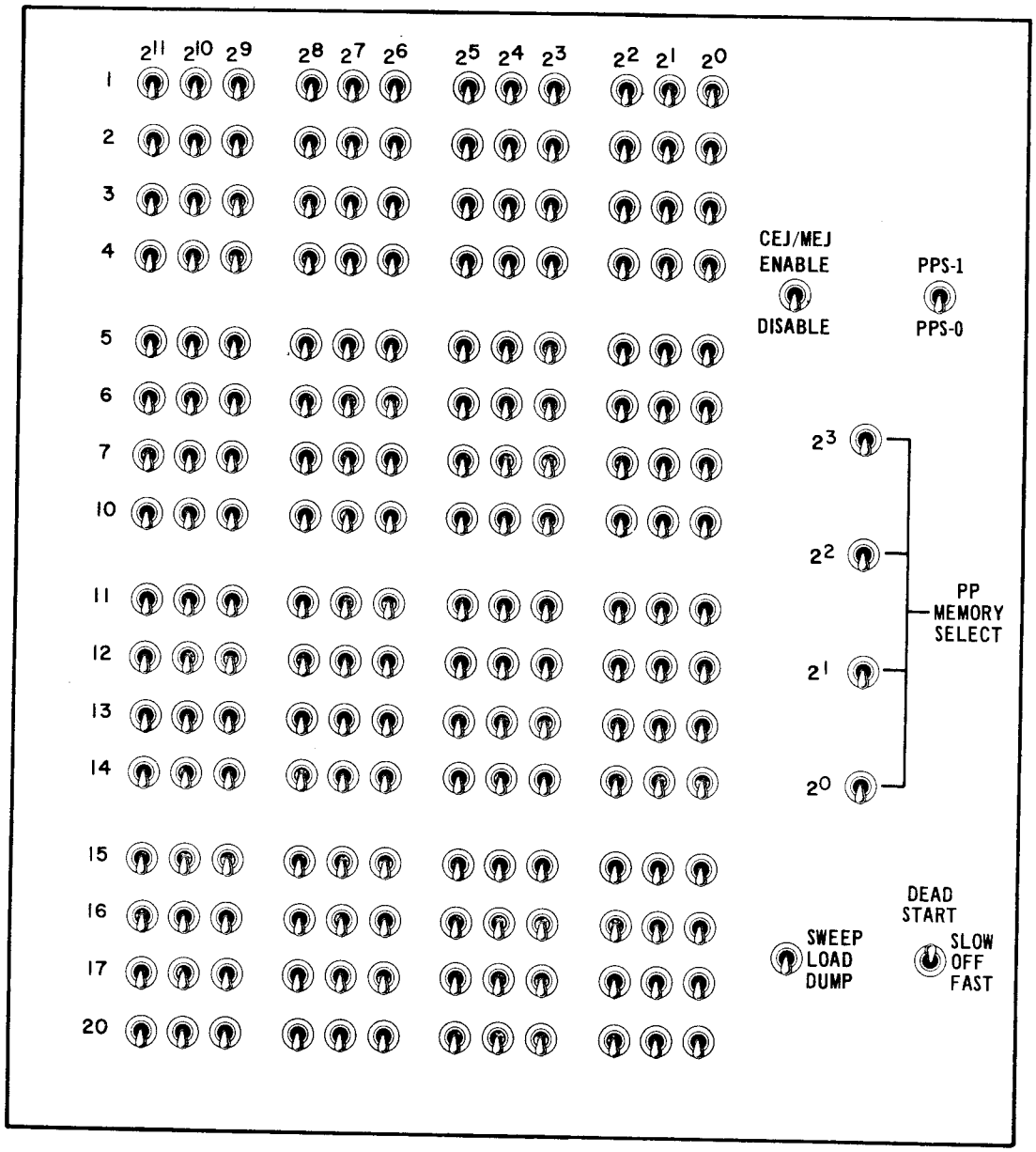
CONTROLS AND INDICATORS — MODELS 171 THROUGH 174

CONTROLS AND INDICATORS — MODELS 171 THROUGH 174

The following descriptions are titled according to the control and indicator functions.

DEADSTART PANEL

The models 171 through 174 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 peripheral processor subsystem (PPS) control switches which are only active during a deadstart. The switches and their functions are listed in table 3-1.



NOTE:

1. DEADSTART PANELS FOR ALL MODELS LOOK ALIKE, EXCEPT FOR THE DEADSTART SWITCH DESIGNATORS SLOW, OFF, AND FAST. THESE DESIGNATORS ARE PRESENT ONLY ON MODELS 171C THROUGH 175C AND 176.

Figure 3-1. Deadstart Panel - Models 171 through 176

TABLE 3-1. DEADSTART PANEL FUNCTIONS - MODELS 171 THROUGH 176

Panel Nomenclature	Description	Function
2^0 through 2^{11} by 1 through 20	Toggle switch matrix (two-position switches)	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 (two-position)	On model 176, this switch is not functional. On models 171 through 175, the switch enables or disables the central exchange jump (CEJ) instruction for the CP and the monitor exchange jump (MEJ) instructions for the peripheral processors (PPs). The switch position is set prior to a deadstart. Resetting the switch after a deadstart does not affect the computer operation until the next deadstart.
PPS-0 PPS-1	Toggle switch (two-position)	Selects PPS-0 and PPS-1 to contain the controlling PP-0. For PP-0 to be in PPS-1, PPS-1 must contain all 10 PPs. Resetting the switch after a deadstart does not affect the computer operation until the next deadstart.
PP MEMORY SELECT $2^3, 2^2, 2^1, 2^0$	Toggle switches (two-position)	Permit the assignment of any peripheral processor memory (PPM) to PP-0. PP-0 has a special control function at deadstart time. If the PPM for PP-0 malfunctions, the user may set the switches to assign any of the other nine PPMs to 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 assignment is made by enabling the switches to form a binary number of the PPM chosen for PP-0 (for example, 0101 selects PPM-5). These switches do not affect the PPS-1 chassis unless that chassis contains all 10 PPs. For software debugging purposes, these switches may be used prior to a deadstart dump to move the logical position of PPM-0 so its contents can be saved. Additional information on this capability is in the NOS Installation Handbook (publication number 60435700) and the NOS System Programmer's Instant (publication number 60449200). Up position sets bit. Down position clears bit.
SWEEP LOAD DUMP	Toggle switch (three-position)	Selects PP-0 mode of operation (refer to Deadstart in this section).
DEAD START (model 171B and models 172 A and B through 175 A and B only)	Toggle switch (two-position)	Provides system deadstart. The deadstart stops the CP. Up position causes deadstart to repeat each 4096 microseconds, which includes a master clear duration of 1.0 microsecond. Down position is deadstart off.
DEAD START (models 171C through 175C and model 176 only)	Toggle switch (three-position)	Provides system deadstart. In models 171C through 175C, the deadstart stops the CP. In model 176, the CP does not stop.
SLOW		Causes deadstart to repeat each 4096 microseconds, which includes a master clear duration of 1.0 microsecond.
OFF		Sets deadstart to off.
FAST		Causes deadstart to repeat each 256 microseconds, which includes a master clear duration of 1.0 microsecond.

I/O CHANNEL PARITY SWITCHES

The models 171 through 174 input/output (I/O) channel parity switches are located on the I/O connector panel (figure 3-2).

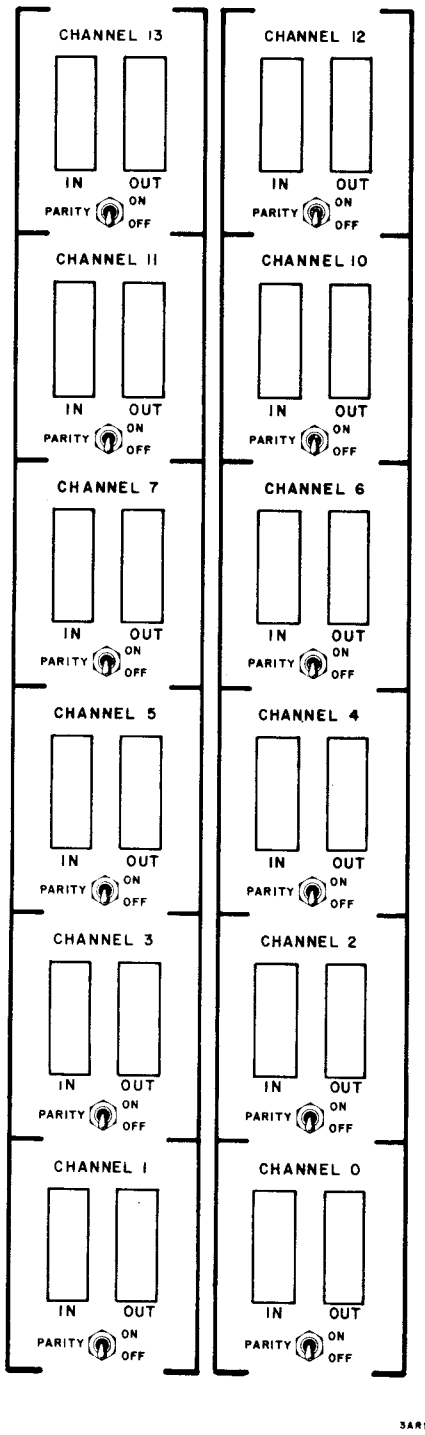


Figure 3-2. Typical I/O Channel Parity Switches for PPS

The panel 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.

ECS PARITY SWITCH

The models 171 through 174 extended core storage (ECS) parity switch is on the module at location 4B34 (figure 3-3).

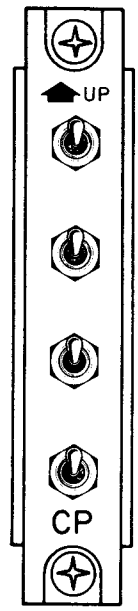


Figure 3-3. Module at 4B34 - Models 171 through 174

This switch 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 UP position. The switch must be in the down position when an ECS controller without parity enhancement is used. The down position may also be used to disable parity enhancement, if it is available but not used.

CLOCK SELECTION SWITCHES AND INDICATORS

The models 171 through 174 clock selection switches are on modules at locations 4R39 and 4R40 (figures 3-4 and 3-5).

The module at 4R39 has one two-position toggle switch, a push-button switch, and three red, light-emitting diode indicators.

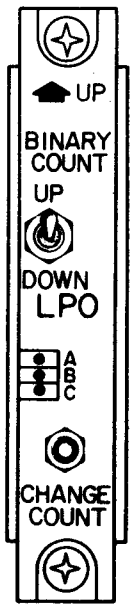


Figure 3-4. Module at 4R39 - Models 171 through 174

The module at 4R40 has two two-position toggle switches.

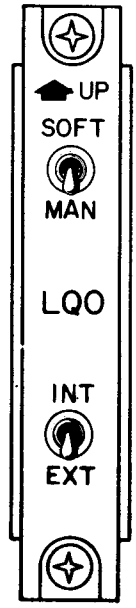


Figure 3-5. Module at 4R40 - Models 171 through 174

Table 3-2 lists the switch and indicator functions of both modules.

TABLE 3-2. FUNCTIONS OF MODULES AT 4R39 AND 4R40 - MODELS 171 THROUGH 174

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 the MAN position. Normal operation requires the clock to be set at the normal operating frequency as indicated by 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 does not light and B and C are lighted. Further use of these indicators is described in the hardware maintenance manuals listed in the system publication index in the preface.
BINARY COUNT UP/DOWN	Toggle switch	Selects an increment or decrement of the binary count, changed with the CHANGE COUNT switch.

CSU MAINTENANCE SWITCHES

The model 171B and models 172 A and B through 174 A and B central storage unit (CSU) maintenance switches are located in chassis 3, and when installed, chassis 7. The switches are on EM type modules at locations 3Q30 and 7Q30 (figure 3-6).

The models 171C through 174C CSU maintenance switches are also located in chassis 3, and when installed, chassis 7. These switches are on QM modules at locations 3I36 and 7I36 (figure 3-6).

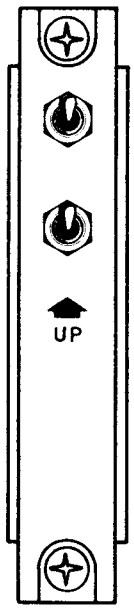


Figure 3-6. Typical EM or QM Module

Table 3-3 lists the switch functions. The switches are always active.

P REGISTER AND STATUS BIT SELECTIOI SWITCHES

The models 171 through 174 program (P) register and status bit selection switches are located in chassis 2 and in chassis 6, when installed. The switches are on module at locations 2J40 and 6J40 (figure 3-7).

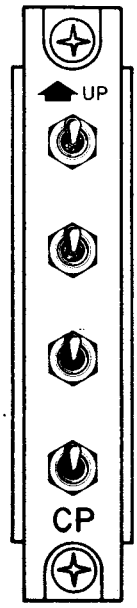


Figure 3-7. Modules at 2J40 and 6J40 - Models 171 through 174

The modules are alike. Each has four two-position toggle switches. The switches permit the selection of the contents of any of the PPS P registers for display on indicators on a module at 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 code with the bottom switch as code bit 0 and the top switch as code bit 3. Table 3-4 lists the switches and their functions.

The switches are active when control bit 124 clears and disabled when bit 124 sets.

TABLE 3-3. CSU MAINTENANCE SWITCH FUNCTIONS

Panel Location	Description	Function
Top	Toggle switch	UP position provides constant master clear in the eight CSU memory banks. A manual master clear results from toggling the switch once from the normal operating position of down.
Middle	Toggle switch	UP position disables address parity detection in the eight CSU memory banks. Down position enables the address parity detection.

TABLE 3-4. FUNCTIONS OF MODULES AT 2J40 AND 6J40 - MODELS 171 THROUGH 174

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

KEYBOARD DISPLAY SELECTION SWITCHES

The models 171 through 174 keyboard display selection switches are on modules at location 2P38 and 2R36 (figure 3-8).

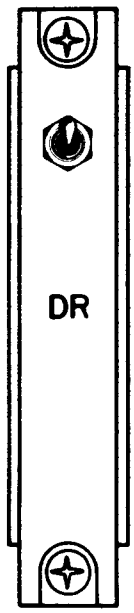


Figure 3-8. Modules at 2P38 and 2R36 - Models 171 through 174

The modules are alike. Each is a keyboard input receiver with one two-position toggle switch. The switch at 2P38 enables (down position) or disables (up position) the keyboard of optional display station 1, if installed. The switch at 2R36 enables (down position) or disables (up position) the keyboard of display station 0. 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.

POWER SENSE MONITOR INDICATORS

The power sense monitor indicators are on models 171B through 174B only. The indicators are on PL and PM type modules in chassis 1, 2, 3, and 4, and when installed, chassis 5, 6, and 7.

The PL and PM modules are for maintenance purposes. They provide indications of dc chassis voltages that exceed the predetermined high or low limits shown in figures 3-9 and 3-10.

The PL modules are at locations 1B42, 2B42, 3C01, 4B42, 5B42, 6B42, and 7C01. Each PL module has four red, light-emitting diodes, a +12-volt calibrating control (R16), and a pushbutton switch for testing the diodes.

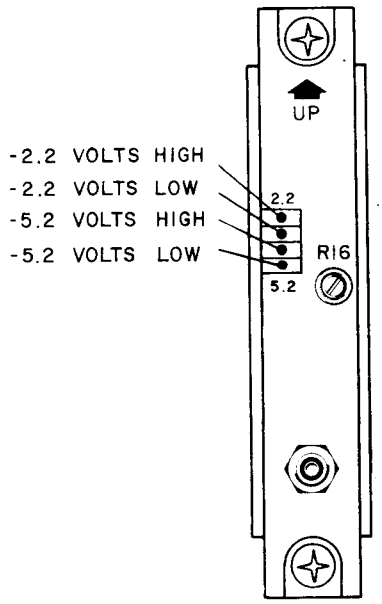


Figure 3-9. PL Module

The PM modules are at locations 2B41, 3C02, 6B41, and 7C02. Each PM module has eight red, light-emitting diodes and a pushbutton switch for testing the diodes.

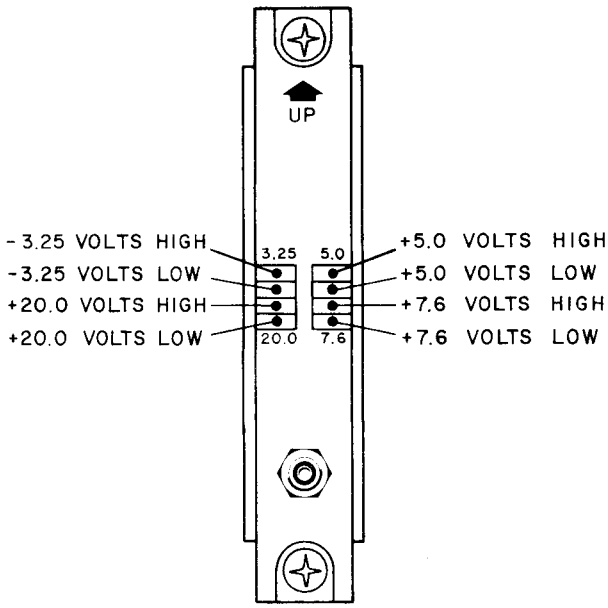


Figure 3-10. PM Module

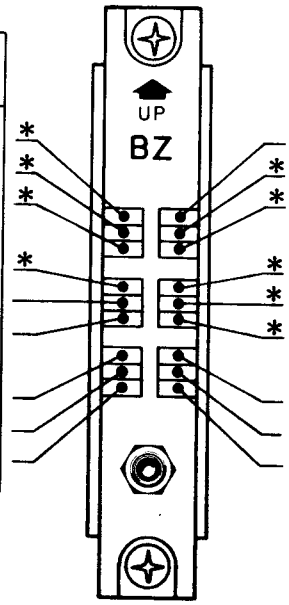
STATUS AND CONTROL REGISTER INDICATORS

The models 171 through 174 status and control register indicators are located in chassis 2 and on chassis 6, when installed. The indicators are on modules at PPS locations F19, F22, F25, F28, F39, F42, and H33 (figures 3-9 through 3-15).

Each module has two columns of nine red, light-emitting diodes. The diodes indicate the condition of certain status and control register bits. The diodes represent bits in the registers and light when their corresponding register bits set. A push-button switch on the module permits testing all the diodes on the module without changing any of the data bits. The light displays are useful in debugging software.

Figures 3-11 through 3-17 show the modules, diode locations (decimal and octal), usage (status or control), and descriptions. The figures apply to the status and control registers of PPS-0 and PPS-1 except where indicated by an asterisk (*), indicating PPS-0 usage only. The light-emitting diodes which are not used but have bit number designations are wired to the status and control register. The diodes that do not have bit designations are not wired.

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
11	13	S	ECS ERROR
12	14	S	CP-0 P RGTR PE
13	15	S	CP-1 P RGTR PE (MODEL 174)
14	16	S	PP0 MEMORY PE
15	17	S	PP1 MEMORY PE
			NOT USED

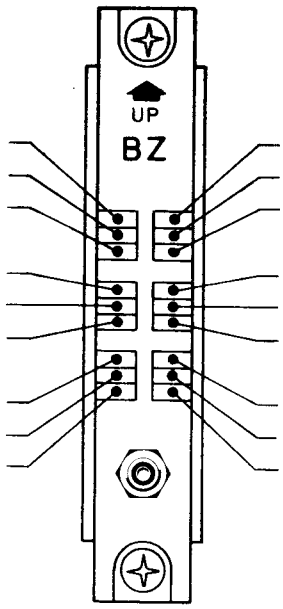


BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
0	0	S	READ PYRAMID PE
1	1	S	CSU-0 ADRS PE
2	2	S	CSU-1 ADRS PE
3	3	S	SECDED ERROR
183	267	S	DOUBLE ERROR
5	5	S	CMC PE
6	6	S	PE FROM EXT CHAN
7	7	S	PE FROM EXT PP
			NOT USED

*BIT APPLIES ONLY TO PPS-0.
BIT 13 APPLIES ONLY TO MODEL 174.

Figure 3-11. Module at PPS F19 - Models 171 through 175

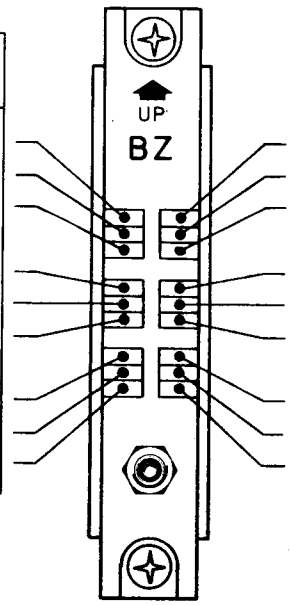
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
24	30	S	CHANNEL 0 PE
25	31	S	CHANNEL 1 PE
26	32	S	CHANNEL 2 PE
27	33	S	CHANNEL 3 PE
28	34	S	CHANNEL 4 PE
29	35	S	CHANNEL 5 PE
30	36	S	CHANNEL 6 PE
31	37	S	CHANNEL 7 PE
11	13	S	LCME RANK II ERROR



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 3-12. Module at PPS F22 - Models 171 through 175

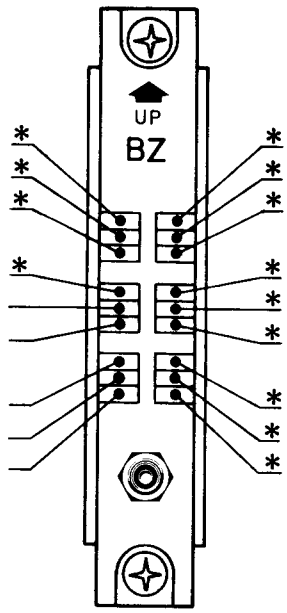
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
40	50	S	SSM SYNDROME BIT 0
41	51	S	SSM SYNDROME BIT 1
42	52	S	SSM SYNDROME BIT 2
43	53	S	SSM SYNDROME BIT 3
44	54	S	SSM SYNDROME BIT 4
45	55	S	SSM SYNDROME BIT 5
46	56	S	SSM SYNDROME BIT 6
47	56	S	SSM SYNDROME BIT 7
			NOT USED



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
183	267	S	DOUBLE ERROR
196	304	S	LCME DOUBLE ERROR
			NOT USED

Figure 3-13. Module at PPS F25 - Models 171 through 175

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

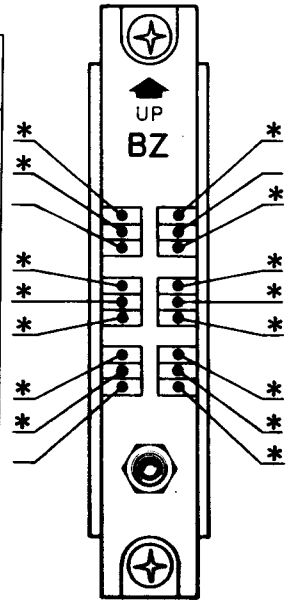


BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
48	60	S	SYNDROME ADRS BIT 0
49	61	S	SYNDROME ADRS BIT 1
50	62	S	SYNDROME ADRS BIT 2
51	63	S	SYNDROME ADRS BIT 15 FOR M172-M174, BIT 16 FOR M175
52	64	S	SYNDROME ADRS BIT 16 FOR M172-M174, BIT 17 FOR M175
53	65	S	SYNDROME ADRS BIT 17 FOR M172-M174, BIT 3 FOR M175
54	66	S	PE PORT CODE BIT 0
55	67	S	PE PORT CODE BIT 1
139	213	S	CMC ADRS/DATA PE

* BIT APPLIES ONLY TO PPS-0.

Figure 3-14. Module at PPS F28 - Models 171 through 175

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

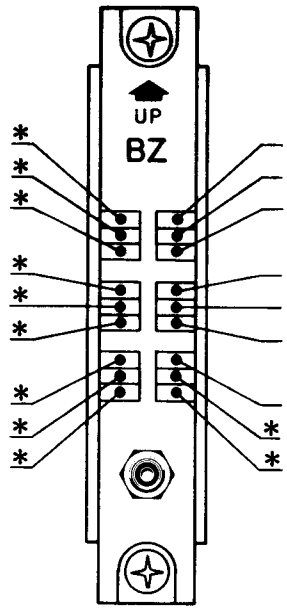


BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
84	124	C	ALL PPs 500 NSEC MAJ CYCLE
85	125	C	INHIBIT CMC REQUEST
86	126		NOT USED
87	127	C	NOT USED
88	128	C	NOT USED
89	129	C	NOT USED
90	130	C	NOT USED
91	131	C	NOT USED
92	132	C	NOT USED

*BIT APPLIES ONLY TO PPS-0.

Figure 3-15. Module at PPS F39 - Models 171 through 175

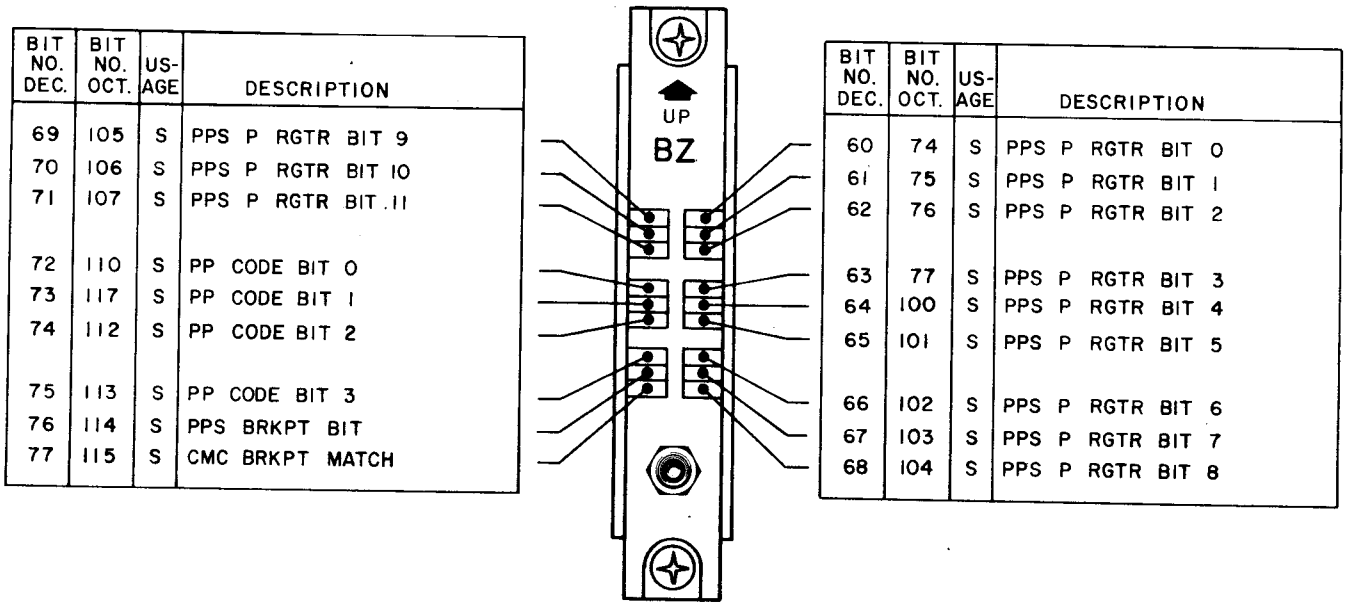
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
144	220	S	RVM ADRS BIT 0 STATUS
145	221	S	RVM ADRS BIT 1 STATUS
146	222	S	RVM ADRS BIT 2 STATUS
147	223	S	RVM ADRS BIT 3 STATUS
148	224	S	RVM ADRS BIT 4 STATUS
149	225	S	RVM ADRS BIT 5 STATUS
150	226	S	RVM ADRS HI/LO
151	227	S	RVM ALL/ONE
			NOT USED



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

* BIT APPLIES ONLY TO PPS-0. BIT 144 THROUGH 151 APPLY ONLY TO MODEL 175.

Figure 3-16. Module at PPS F42 - Models 171 through 175



BIT NO. DEC.	BIT NO. OCT.	USAGE	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.	USAGE	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

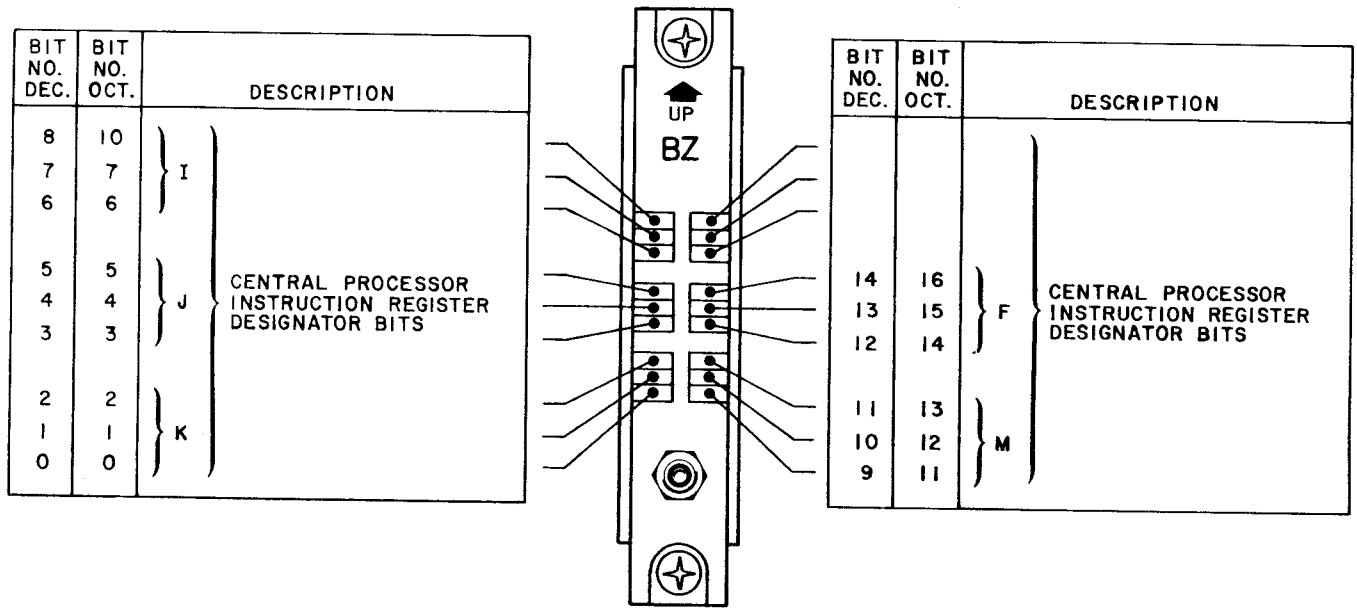
Figure 3-17. Module at PPS H33 - Models 171 through 175

CPU INSTRUCTION REGISTER AND P REGISTER INDICATORS

The models 171 through 174 central processing unit (CPU) instruction register and P register indicators are on two modules (figures 3-18 and 3-19). In models 171 through 173, the modules are at locations 1C21 and 1K38. In model 174, the modules are at locations 5C21 and 5K38.

Each module has two columns of nine red, light-emitting diodes which display the content of the CPU instruction register at respective module locations 1C21 and 5C21 and the content of the CPU P register at respective module locations 1K38 and 5K38. Each module diode represents one register bit and lights when the bit sets. A push-button switch below the diodes permits testing the diodes on the module without changing any data.

The module displays are useful in debugging software.



BIT NO. DEC.	BIT NO. OCT.		DESCRIPTION
8	10	I	CENTRAL PROCESSOR INSTRUCTION REGISTER DESIGNATOR BITS
7	7	I	
6	6	I	
5	5	J	
4	4	J	
3	3	J	
2	2	K	
1	1	K	
0	0	K	

BIT NO. DEC.	BIT NO. OCT.		DESCRIPTION
14	16	F	CENTRAL PROCESSOR INSTRUCTION REGISTER DESIGNATOR BITS
13	15	F	
12	14	F	
11	13	M	
10	12	M	
9	11	M	

Figure 3-18. Modules at 1C21 - Models 171 through 173 and at 5C21 - Model 174

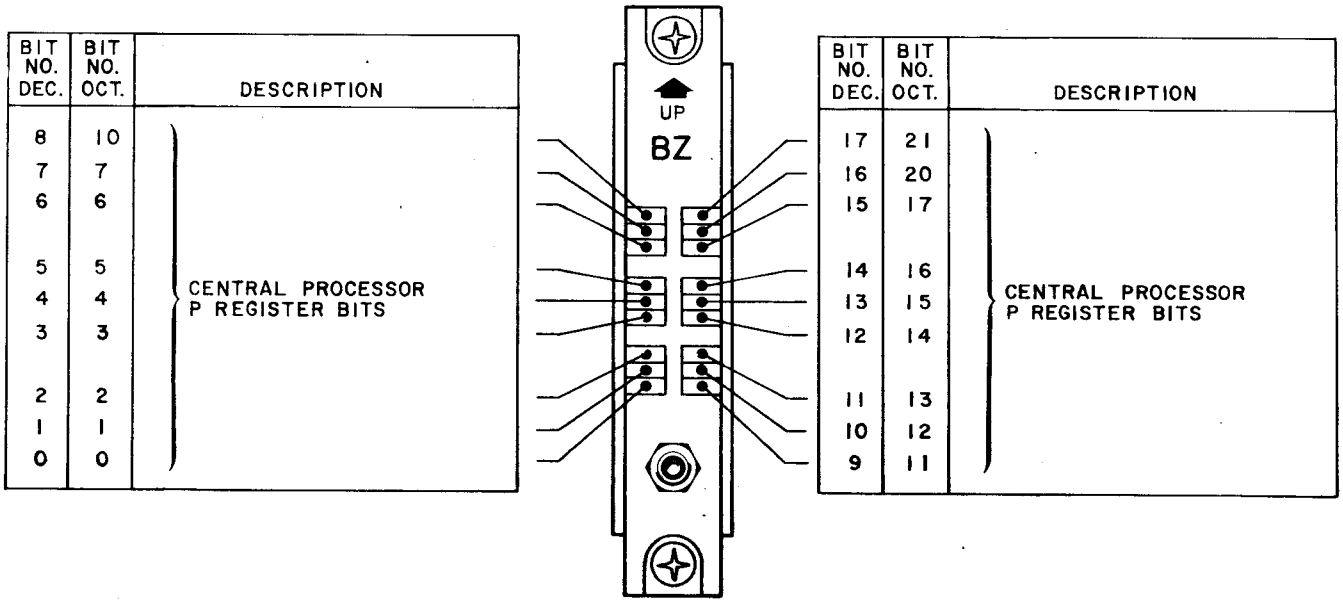


Figure 3-19. Modules at 1K38 - Models 171 through 173 and at 5K38 - Model 174

CM CONFIGURATION AND SECDED/PARITY MODE SWITCHES

The models 171 through 174 central memory (CM) configuration and SECDED/parity mode switches are on module at locations 4N24 and 4N25 (figure 3-20).

Tables 3-5 and 3-6 list the module switches and their functions.

All CM quadrants, for a particular CM size, are available for use by model 171 through 174 systems when the address range control and bad quadrant code switches are set for normal operation as shown in table 3-7.

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 3-7. 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.

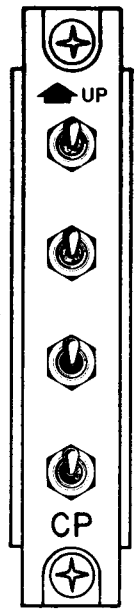


Figure 3-20. Modules at 4N24 and 4N25 - Models 171 through 174

TABLE 3-5. FUNCTIONS OF MODULE AT 4N24 - MODELS 171 THROUGH 174

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 3-6. FUNCTIONS OF MODULE AT 4N25 - MODELS 171 THROUGH 174

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 3-7. MEMORY SELECTION SCHEME - MODELS 171 THROUGH 174

Central Memory Size	Range of Address	Normal Operation Switch Positions†							Reconfigured Operation Switch Positions†						
		Address Range Control Switch				Bad Quadrant Code Switch			Reconfigured Selection		Address Range Control Switch				Code Switch
		1	2	3	4	5	6	7	CSU	Bad Quadrant	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.

CONTROLS AND INDICATORS — MODEL 175

CONTROLS AND INDICATORS — MODEL 175

The following descriptions are titled according to the control and indicator functions. Some of these functions are the same as those for models 171 through 174. In these instances, references are made to the corresponding control and indicator descriptions for models 171 through 174.

DEADSTART PANEL

For the model 175 deadstart panel description, refer to the description for models 171 through 174 in this section.

I/O CHANNEL PARITY SWITCHES

For the model 175 I/O channel parity switches description, refer to the description for models 171 through 174 in this section.

ECS PARITY SWITCH

For the model 175 ECS parity switch description, refer to the description for models 171 through 174 in this section, except use module location 4C35 in place of 4B34.

CLOCK SELECTION SWITCHES AND INDICATORS

For models 175 A and B, refer to the description for models 171 through 174 in this section, except use module locations 4B18 and 4B19 in place of 4R39 and 4R40.

For model 175C, refer to the description for models 171 through 174 in this section, except use module locations 4D36 and 4D37 in place of 4R39 and 4R40.

CSU MAINTENANCE SWITCHES

The model 171B and models 172 A and B through 174 A and B central storage unit (CSU) maintenance switches are located in chassis 3, and when installed, chassis 7. The switches are on EM type modules at locations 3Q30 and 7Q30 (figure 3-6).

The models 171C through 174C CSU maintenance switches are also located in chassis 3, and when installed, chassis 7. These switches are on QM modules at locations 3I36 and 7I36 (figure 3-6).

P REGISTER AND STATUS BIT SELECTION SWITCHES

For the model 175 P register and status bit selection description, refer to the description for models 171 through 174 in this section.

KEYBOARD DISPLAY SELECTION SWITCHES

For the model 175 keyboard selection switches description, refer to the description for models 171 through 174 in this section.

POWER SENSE MONITOR INDICATORS

The power sense monitor indicators are on model 175B only. The indicators are on PL and PM type modules in chassis 1, 2, 3, and 4, and when installed, chassis 10. For a description of the modules, refer to the models 171B through 174B descriptions, except use chassis locations 1C01, 2B42, 3C01, 4B42, and 10B42 for the PL module. Use chassis locations 1C02, 2B41, 3C02, and 10B41 for the PM module.

STATUS AND CONTROL REGISTER INDICATORS

For the model 175 status and control indicators descriptions, refer to the description for models 171 through 174 in this section.

CM CONFIGURATION AND CLOCK SWITCHES AND INDICATORS

The model 175 CP contains clock switches and indicators at module locations 5A1 through 5A3 (figure 3-21). Table 3-8 lists the switches, indicators, and their functions. Table 3-9 lists the switch settings for a normal or reconfigured CM.

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 3-9.

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 3-9. 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.

Any errors detected while CM is operating in a reconfigured (degraded) mode appear to the status and control register in translated form, giving the physical address (bank, quadrant, and so on) of the error.

The switches are always active.

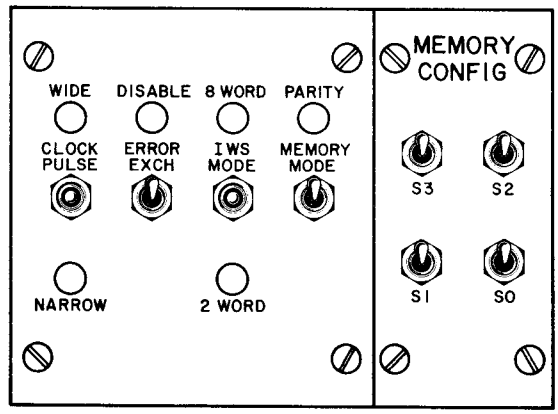


Figure 3-21. Controls on Modules at 5A1 through 5A3 - Model 175

TABLE 3-8. FUNCTIONS OF CONTROLS ON MODULES AT 5A1 THROUGH 5A3 - MODEL 175

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 disabled 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.

TABLE 3-9. MEMORY SELECTION SCHEME - MODEL 175

Central Memory Size	Range of Address	Normal Operation Switch Positions†				Reconfigured Operation Switch Positions†				
		Memory Configuration Switches				Bad Quadrant	Memory Configuration Switches			
		S0	S1	S2	S3		S0	S1	S2	S3
65K	0-1777777	1	0	0	0	0	No reconfiguration			
98K	0-2777777	1	1	0	0	0	No reconfiguration			
						1	1	0	0	0
131K	0-3777777	1	1	0	0	0	0	1	0	0
						1	1	0	0	0
196K	0-5777777	1	1	1	0	0	0	1	1	0
						1	1	0	1	0
						2	1	1	0	0
262K	0-7777777	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.

CONTROLS AND INDICATORS — MODEL 176

CONTROLS AND INDICATORS — MODEL 176

The following descriptions are titled according to the control and indicator functions. Some of these functions are the same as those for models 171 through 174. In these instances, references are made to the corresponding control and indicator descriptions for models 171 through 174.

DEADSTART PANEL

For the model 176 deadstart panel description, refer to the description for models 171 through 174, except for the panel location. The deadstart panel for model 176 is in the stand-alone cabinet to the right of chassis 2, as seen when facing chassis 2.

I/O CHANNEL PARITY SWITCHES

For the model 176 I/O channel parity switches description, refer to the description for models 171 through 174, except for the I/O connector panel location. The panel for model 176 is in the stand-alone cabinet to the left of chassis 2, as seen when facing chassis 2.

P REGISTER AND STATUS BIT SELECTION SWITCHES

For the model 176 P register and status bit selection switches descriptions, refer to the description for models 171 through 174, except for the location of one switch module. The module for model 176 is located at 4J40 instead of 6J40.

KEYBOARD DISPLAY SELECTION SWITCHES

For the model 176 keyboard selection switches description, refer to the description of models 171 through 174 in this section.

CP CLOCK FREQUENCY SELECTION SWITCHES AND INDICATORS

The model 176 clock frequency selection switches and indicators are at module location 7M06 (figure 3-22). The module includes two three-position toggle switches and four indicator lights. Table 3-10 lists the switch and indicator functions.

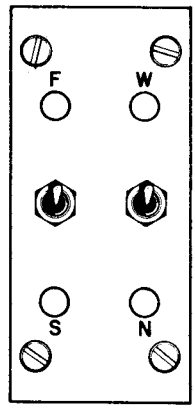


Figure 3-22. Switches and Indicators on Module at 7M06 - Model 176

TABLE 3-10. FUNCTIONS OF SWITCHES AND INDICATORS ON MODULE AT 7M06 - MODEL 176

Panel Nomenclature	Description	Function
Switch between F and S indicators	Toggle switch	Controls clock operating frequency. Up position provides fast clock frequency. Middle position enables software control of clock frequency. Down position provides slow clock frequency.
F, S	Indicators	Light to show respective clock operating frequency.
Switch between W and N indicators	Toggle switch	Controls clock pulse width. Up position provides wide pulse. Middle position enables software control of pulse width. Down position provides narrow pulse.
W, N	Indicators	Light to show respective clock pulse widths.

PPS CLOCK FREQUENCY SELECTION SWITCH

The model 176 clock frequency selection switch is on the module at location 2R29 (figure 3-23).

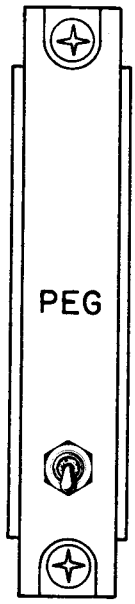


Figure 3-23. Module at 2R29 - Model 176

The switch enables the setting of clock frequency margins of minus four percent (up position), plus four percent (down position), and normal clock (center position). The switch is always active.

CM CONFIGURATION SWITCHES

The model 176 CP contains four CM configuration toggle switches at module location 8K14 (figure 3-24). The switches are labeled S0 through S3. These switches control the CM quadrant configurations by setting selection bits. Table 3-11 shows the switch settings for a normal or reconfigured CM.

All CM quadrants, for a particular CM size, are available for use by the model 176 system when the CM configuration switches are set to the normal operation positions shown in table 3-11.

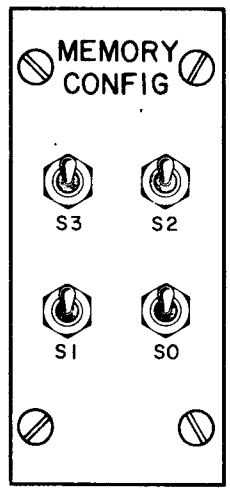


Figure 3-24. Switches on Module at 8K14 - Model 176

TABLE 3-11. MEMORY SELECTION SCHEME - MODEL 176

Central Memory Size	Range of Address	Normal Operation Switch Positions†				Reconfigured Operation Switch Positions†				
		Memory Configuration Switches				Bad Quadrant	Memory Configuration Switches			
		S0	S1	S2	S3		S1	S2	S3	S4
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.

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 3-11. Any one quadrant may be reconfigured at one time, except quadrant 0 of the 65K memory. The switches accomplish a logical reconfiguration by manipulating the CM upper address bits.

Any errors detected while CM is operating in a reconfigured (degraded) mode appear to the status and control register in translated form, giving the physical address (bank, quadrant, and so on) of the error.

The switches are always active.

LCME BANK SELECTION SWITCHES

The model 176 LCME bank selection switches are at module locations 5H14 and 5H15 (figure 3-25). The switches are rotary types with eight positions, numbered 0 through 7. The switches select the LCME size and configuration as shown in table 3-12.

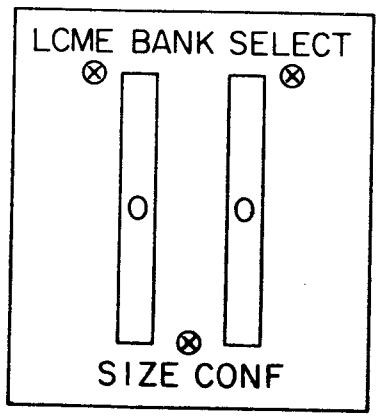


Figure 3-25. Switches on Modules at 5H14 and 5H15 - Model 176

TABLE 3-12. FUNCTIONS OF SWITCHES ON MODULES AT 5H14 AND 5H15

SIZE†	CONF††	Selected Memory Size	Selected Banks
0	0	512K	0-1
0	1	512K	2-3
1	0	1024K	0-3
1	1	1024K	4-7
2	0	2048K	0-7
5, 6, 7	These positions permit software control of the LCME size through the status and control register.		

† Positions 3 and 4 are unused.
†† Positions 2 through 7 are unused.

POWER SENSE MONITOR INDICATORS

The model 176 power sense monitor indicators are on PL and PM type modules in chassis 2, and when installed, chassis 4. For a description of the modules, refer to the models 171B through 174B descriptions, except use chassis locations 2B42 and 4B42 for the PL module. Use chassis locations 2B41 and 4B41 for the PM module locations.

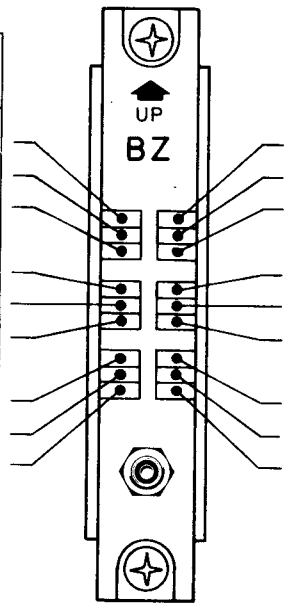
PPS-0 STATUS AND CONTROL REGISTER INDICATORS

The model 176 status and control register indicators for PPS-0 are on modules at locations 2F19, 2F22, 2F25, 2F28, 2F39, 2F42, and 2H33 (figures 3-26 through 3-32).

Each module has two columns of nine red, light-emitting diodes. The diodes indicate the condition of certain status and control register bits. Each diode represents a bit in the register and lights when the bit sets. A push-button switch on the module permits testing all of the diodes on the module without changing any of the data bits. The light displays are useful in debugging software.

Figures 3-26 through 3-32 show the modules, diode locations (decimal and octal), usage (status or control), and descriptions for PPS-0. The light-emitting diodes which are not used but have bit number designations are wired to the status and control register. The diodes without bit designators are not wired.

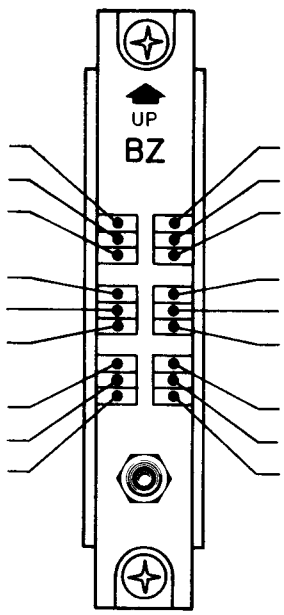
BIT NO. DEC.	BIT NO. OCT.	USAGE	DESCRIPTION
190	276	S	FUTURE LCME USE
191	277	S	FUTURE LCME USE
192	300	S	LCME DEGRADE BIT 0
193	301	S	LCME DEGRADE BIT 1
194	302	S	LCME DEGRADE BIT 2
195	303	S	LCME DEGRADE BIT 3
14	16	S	PP0 MEMORY PE
15	17	S	PP1 MEMORY PE
4	4	S	PPU ERROR



BIT NO. DEC.	BIT NO. OCT.	USAGE	DESCRIPTION
136	210	S	PPU PROGRAM ERROR
132	204	S	PPU PE STACK 0
133	205	S	PPU PE STACK 1
134	206	S	PPU PE STACK 2
135	207	S	PPU PE STACK 3
10	12	S	ERROR IN SECOND PPS
6	6	S	PE FROM EXT CHAN
7	7	S	PE FROM EXT PP
3	3	S	SSM RANK II ERROR

Figure 3-26. Module at 2F19 - Model 176

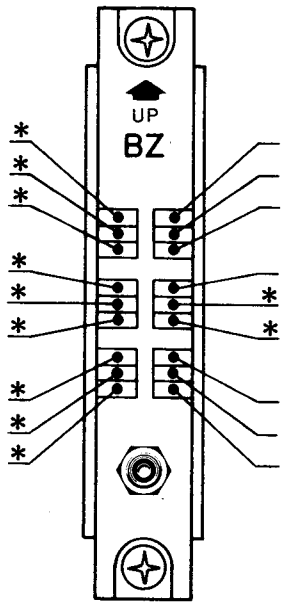
BIT NO. DEC.	BIT NO. OCT.	USAGE	DESCRIPTION
24	30	S	CHANNEL 0 PE
25	31	S	CHANNEL 1 PE
26	32	S	CHANNEL 2 PE
27	33	S	CHANNEL 3 PE
28	34	S	CHANNEL 4 PE
29	35	S	CHANNEL 5 PE
30	36	S	CHANNEL 6 PE
31	37	S	CHANNEL 7 PE
			NOT USED



BIT NO. DEC.	BIT NO. OCT.	USAGE	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 3-27. Module at 2F22 - Model 176

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

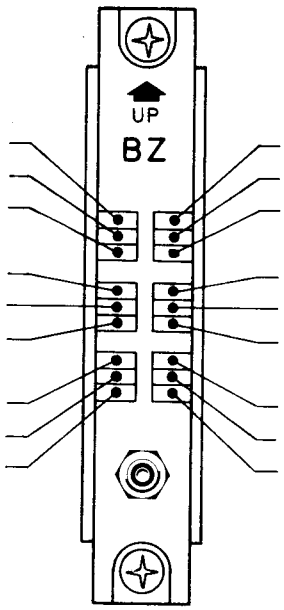


BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
32	40	S	CHANNEL 10 PE
33	41	S	CHANNEL 11 PE
34	42	S	CHANNEL 12 PE
35	43	S	CHANNEL 13 PE
36	44	S	MAINS POWER FAILURE
37	45	S	SHUTDOWN IMMINENT
38	46		NOT USED
39	47		NOT USED
			NOT USED

* BIT APPLIES ONLY TO PPS-0.

Figure 3-28. Module at 2F25 - Model 176

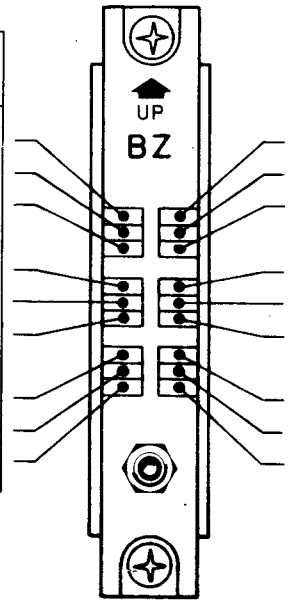
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
179	263	C	CHANNELS 2, 3 BIAS BIT 0
180	264	C	CHANNELS 2, 3 BIAS BIT 1
181	265	C	CHANNELS 2, 3 BIAS BIT 2
182	266	C	CHANNELS 2, 3 BIAS BIT 3
184	270	C	CHANNELS 4-7 BIAS BIT 0
185	271	C	CHANNELS 4-7 BIAS BIT 1
186	272	C	CHANNELS 4-7 BIAS BIT 2
187	273	C	CHANNELS 4-7 BIAS BIT 3
			NOT USED



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
54	66	C	EXCH BFR BIAS BIT 0
55	67	C	EXCH BFR BIAS BIT 1
56	68	C	EXCH BFR BIAS BIT 2
57	69	C	EXCH BFR BIAS BIT 3
72	110	C	SCANNER SELECT BIT 0
73	111	C	SCANNER SELECT BIT 1
74	112	C	SCANNER SELECT BIT 2
75	113	C	SCANNER SELECT BIT 3
82	122	C	ENABLE SCAN INTERFACE

Figure 3-29. Module at 2F28 - Model 176

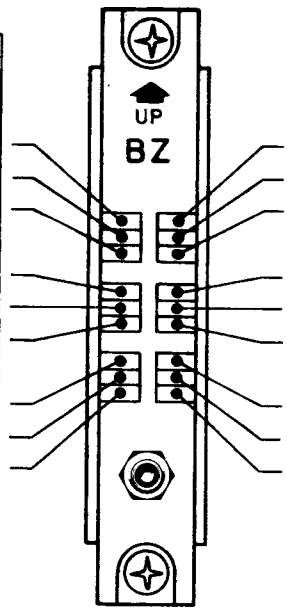
BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
161	241	S	CM ERROR ADRS BIT 9
162	242	S	CM ERROR ADRS BIT 10
163	243	S	CM ERROR ADRS BIT 11
164	244	S	CM ERROR ADRS BIT 12
165	245	S	CM ERROR ADRS BIT 13
166	246	S	CM ERROR ADRS BIT 14
167	247	S	CM ERROR ADRS BIT 15
48	60	S	CM ERROR ADRS BIT 16
49	61	S	CM ERROR ADRS BIT 17



BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
50	62	S	CM ERROR ADRS BIT 0
51	63	S	CM ERROR ADRS BIT 1
52	64	S	CM ERROR ADRS BIT 2
53	65	S	CM ERROR ADRS BIT 3
156	234	S	CM ERROR ADRS BIT 4
157	235	S	CM ERROR ADRS BIT 5
158	236	S	CM ERROR ADRS BIT 6
159	237	S	CM ERROR ADRS BIT 7
160	240	S	CM ERROR ADRS BIT 8

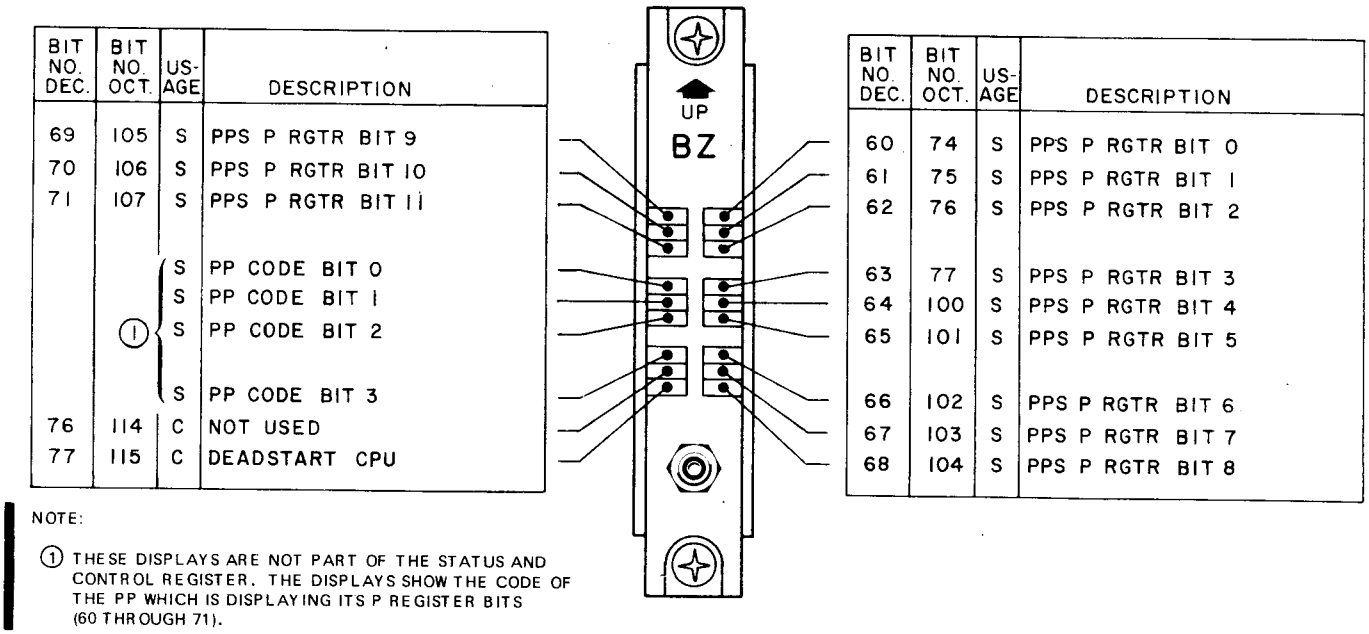
Figure 3-30. Module at 2F39 - Model 176

BIT NO. DEC.	BIT NO. OCT.	US-AGE	DESCRIPTION
144	220	S	LCME SYNDROME BIT 0
145	221	S	LCME SYNDROME BIT 1
146	222	S	LCME SYNDROME BIT 2
147	223	S	LCME SYNDROME BIT 3
148	224	S	LCME SYNDROME BIT 4
149	225	S	LCME SYNDROME BIT 5
150	226	S	LCME SYNDROME BIT 6
151	227	S	LCME SYNDROME BIT 7
86	126	C	BLOCK COPY CONTROL



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
84	124	C	ALL PPs 500NSEC MAJ CYCLE
85	125	C	INHIBIT CMC REQUEST
95	137	C	STOP ON PP MEMORY PE
143	217	S	CLOCK MARGIN

Figure 3-31. Module at 2F42 - Model 176



BIT NO. DEC.	BIT NO. OCT.	USAGE	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
		S	PP CODE BIT 0
		S	PP CODE BIT 1
	(1)	S	PP CODE BIT 2
		S	PP CODE BIT 3
76	114	C	NOT USED
77	115	C	DEADSTART CPU

BIT NO. DEC.	BIT NO. OCT.	USAGE	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

NOTE:

(1) THESE DISPLAYS ARE NOT PART OF THE STATUS AND CONTROL REGISTER. THE DISPLAYS SHOW THE CODE OF THE PP WHICH IS DISPLAYING ITS P REGISTER BITS (60 THROUGH 71).

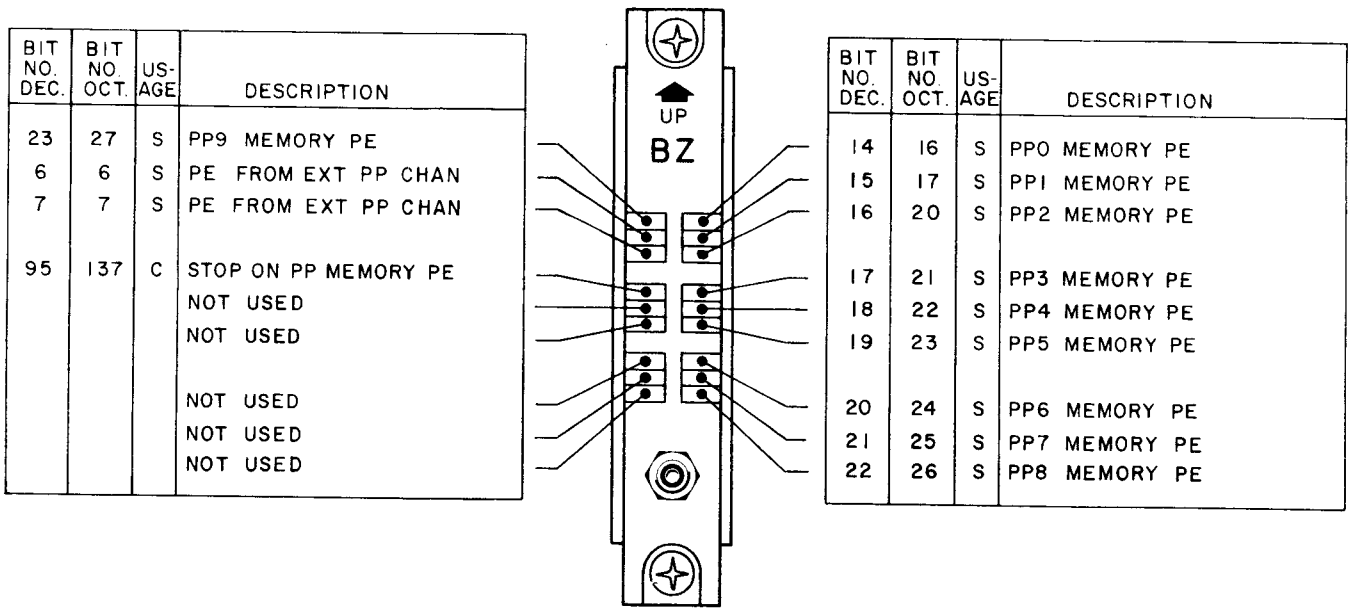
Figure 3-32. Module at 2H33 - Model 176

PPS-1 STATUS AND CONTROL REGISTER INDICATORS

The model 176 status and control register indicators for PPS-1 are located in chassis 4, when installed. The indicators are on module at locations 4T22, 4F25, and 4H33 (figures 3-33 through 3-35).

Each module has two columns of nine red, light-emitting diodes. The diodes indicate the condition of certain status and control register bits. Each diode represents a bit in the register and lights when the bit sets. A push-button switch on the module permits testing all of the diodes on the module without changing any of the data bits. The light displays are useful in debugging software.

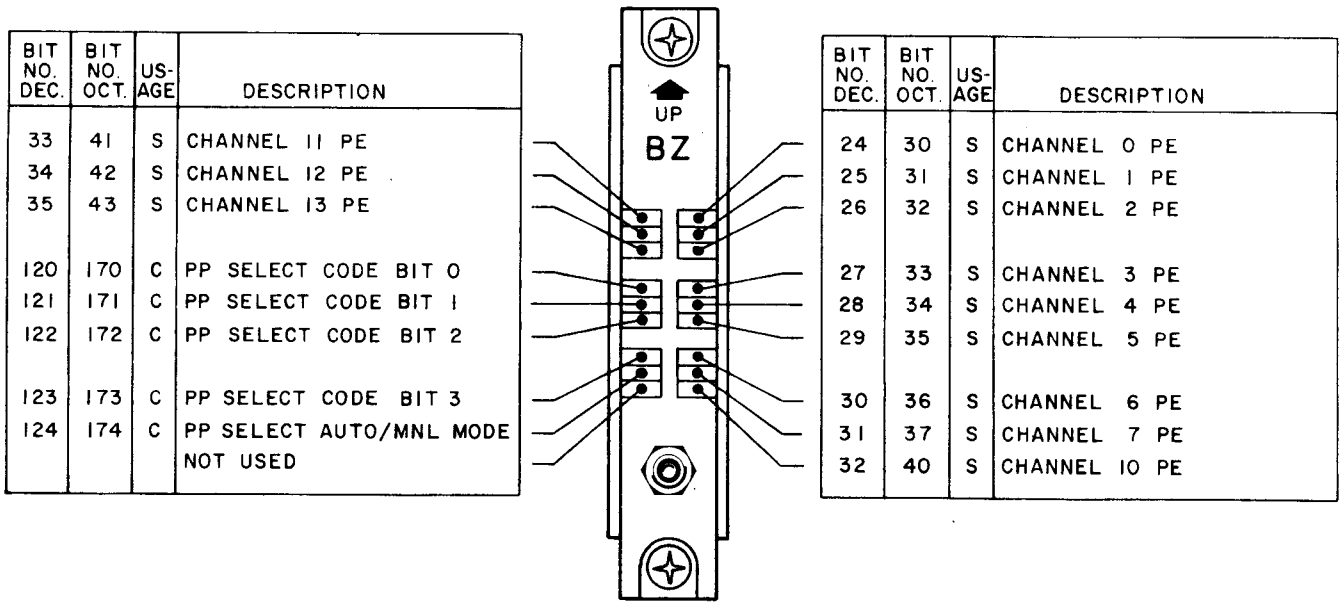
Figures 3-33 and 3-35 show the modules, diode locations (decimal and octal), usage (status or control), and descriptions for PPS-1. The light-emitting diodes which are not used but have bit number designations are wired to the status and control register. The diodes without bit designators are not wired.



BIT NO. DEC.	BIT NO. OCT.	USAGE	DESCRIPTION
23	27	S	PP9 MEMORY PE
6	6	S	PE FROM EXT PP CHAN
7	7	S	PE FROM EXT PP CHAN
95	137	C	STOP ON PP MEMORY PE
			NOT USED
			NOT USED
			NOT USED
			NOT USED
			NOT USED

BIT NO. DEC.	BIT NO. OCT.	USAGE	DESCRIPTION
14	16	S	PP0 MEMORY PE
15	17	S	PP1 MEMORY PE
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

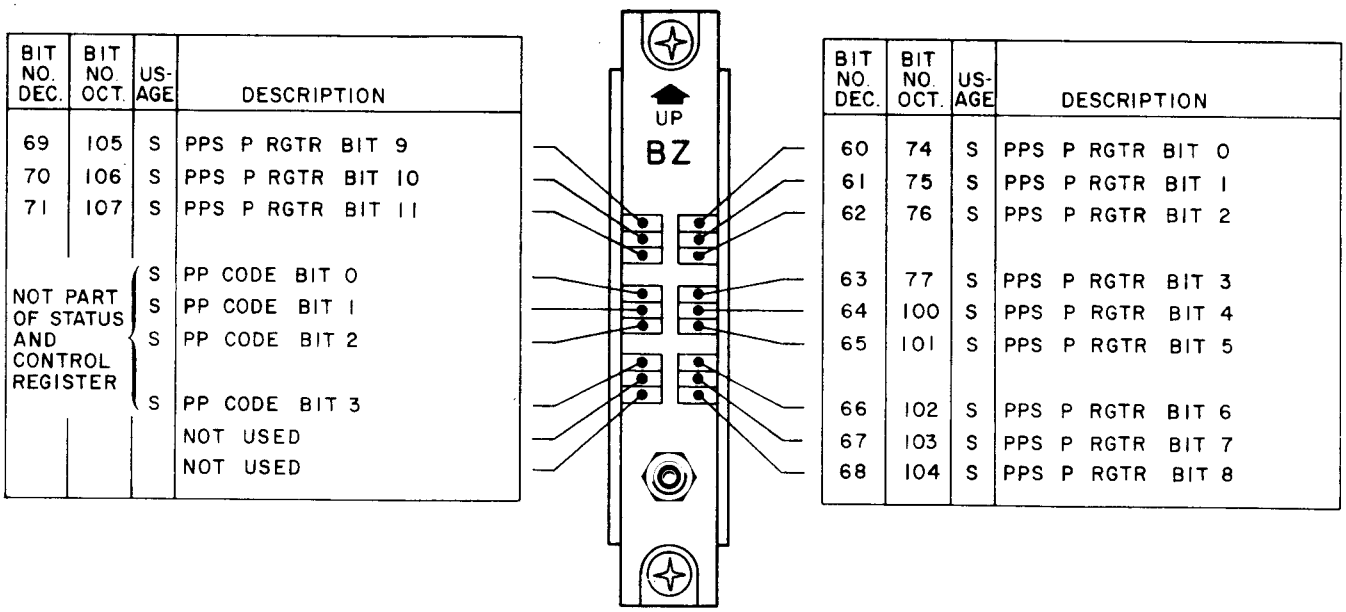
Figure 3-33. Module at 4F22 - Model 176



BIT NO. DEC.	BIT NO. OCT.	USAGE	DESCRIPTION
33	41	S	CHANNEL 11 PE
34	42	S	CHANNEL 12 PE
35	43	S	CHANNEL 13 PE
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

BIT NO. DEC.	BIT NO. OCT.	USAGE	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
32	40	S	CHANNEL 10 PE

Figure 3-34. Module at 4F25 - Model 176



BIT NO. DEC.	BIT NO. OCT.	USAGE	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
NOT PART OF STATUS AND CONTROL REGISTER		S	PP CODE BIT 0
		S	PP CODE BIT 1
		S	PP CODE BIT 2
		S	PP CODE BIT 3
			NOT USED
			NOT USED

BIT NO. DEC.	BIT NO. OCT.	USAGE	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 3-35. Module at 4H33 - Model 176

POWER-ON AND POWER-OFF PROCEDURES — MODELS 171 THROUGH 176

OPERATING PROCEDURES — MODELS 171 THROUGH 176

POWER-ON AND POWER-OFF PROCEDURES — MODELS 171 THROUGH 176

Circuitry within the CDC CYBER 170 systems may be damaged with improper application or removal of power. The power-on and power-off procedures are to be used by designated personnel only and are not included here. These procedures must be referenced in the Power Distribution and Warning System Manual listed in the system publication index in the preface.

CAUTION

Computer operators are generally restricted from using the power-on and power-off controls, except for the system EMERGENCY OFF switch.

OPERATING PROCEDURES — MODELS 171 THROUGH 176

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

Before deadstarting any of the CDC CYBER 170 systems, the positions of the control switches should be checked against their intended use. These checks can most easily be made through the use of the applicable control and indicator descriptions in this section.

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 mode switch to LOAD position.
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 and MEMORY SELECT switches select the desired PP to be PP-0.
4. Set DEAD START switch to UP position, momentarily.

Results of steps 1 through 4:

1. Assign data channels 0 through 11 (octal) to corresponding PPs in each PPS.
2. Send a master clear (MC) to all I/O channels. An MC selects all channel converters on each channel and resets bits 80 through 95 and 120 through 127 of the status and control register. MC sets all channels to the active and empty condition, ready for inputs.
3. Set all PPs to the input (710) instruction.
4. Clear the P registers and set the A registers to 10000 (octal) in all PPs.
5. Transmit a zero word that is followed by the 16 words from the toggle switches into PPM locations 0 through 20 (octal) of PP-0. Channel 0 is then disconnected, clearing word 21 (octal) of PP-0 and causing PP-0 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 mode switch to SWEEP position.
2. Set DEAD START switch to ON position momentarily. (DEAD START switch may be left on for synchronizing 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 contents of each location, without executing instructions.

Dump Mode

Dump mode provides copying the contents of a PPM to an external storage device. Initiate dump program in PP-0 as follows:

1. Set mode switch to DUMP position.

2. Set DEAD START switch to ON position momentarily.

Results of steps 1 and 2:

1. All PPs set to output (730) instruction.

2. An MC is sent on all channels.

3. Channel 0 is held active and empty.

4. Each PP is assigned to its corresponding I/O channel.

5. All P registers clear and all A registers set to 10000 (octal).

6. All PPs sense the empty and active conditions of their assigned channels, output the contents of their adress 0000, set their channels to full, and wait for an empty condition.

7. All PPs advance P by 1 and reduce the A register by 1 (A equals 777776).

8. Because channel 0 is held active and empty, PP0 cycles through the 730 instruction until A equals 1.

9. PP-0 goes to memory location 0001 for its next instruction. PP-0 can send its entire memory contents on channel 0 although no I/O device has been selected to receive it. PP-0 is then free to execute a dump program which must previously have been stored in memory, beginning at location 0001.

This section describes the central processor (CP), peripheral processor unit (PPU), 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. The PPU instructions apply only to model 176. The PP instructions are identical for all models. Instruction timing information follows respective instruction descriptions. (Other programming information and system error response information are in section 5.)

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

CENTRAL PROCESSOR INSTRUCTIONS — MODELS 171 THROUGH 176

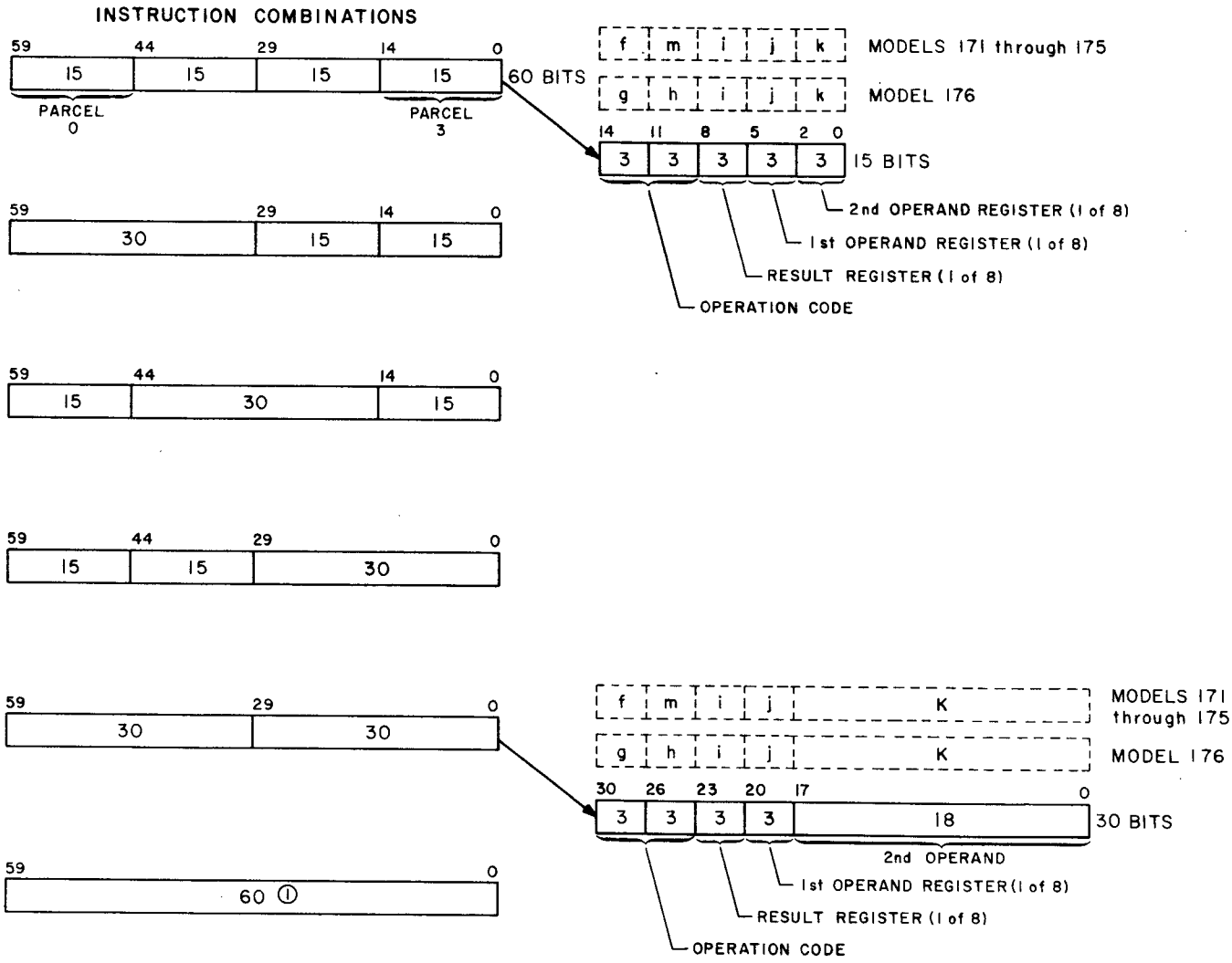
CENTRAL PROCESSOR INSTRUCTIONS — MODELS 171 THROUGH 176

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 follows each instruction set. The timing information allows a complete analysis of the situations warranting the programming effort.

CP 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. Figure 4-1 shows possible parcel arrangements for instructions within a program instruction word.

In models 171 through 175, an instruction may occupy one, two, or four parcels. This arrangement depends upon the instruction format. When an instruction occupies two parcels, it must occupy two parcels within the same program word.



NOTE:

① FOR MODEL 175, REFER TO COMPARE/MOVE INSTRUCTION FORMAT SHOWN WITH THE MOVE DIRECT (465) INSTRUCTION DESCRIPTION. A 60-BIT INSTRUCTION DOES NOT APPLY TO MODEL 176.

3AR7C

Figure 4-1. CP Instruction Parcel Arrangement

In model 176, an instruction may occupy one or two parcels, depending upon the instruction format. If a two-parcel instruction begins in parcel 3 (last parcel) of an instruction word, the instruction does not continue in the following word. The instruction executes as if the instruction word contains a fifth parcel and the fifth parcel contains all zeros.

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 460xx through 463xx for models 171 through 175 and 46xxx for model 176. Instructions 60xxx through 62xxx may be used as two-parcel pass instructions by setting the i instruction designator to zero. Refer to table 4-1 for CP instruction designators.

TABLE 4-1. CENTRAL PROCESSOR INSTRUCTION DESIGNATORS

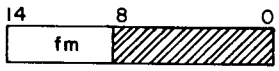
Model/Designator			Use
171 through 174	175	176	
fm	fm	gh	6-bit instruction code
fmi	fmi	ghi	9-bit instruction code
i	i	i	3-bit code specifying one of eight registers
j	j	j	3-bit code specifying one of eight registers
jk	jk	jk	6-bit code specifying amount of shift or mask
k	k	k	3-bit code specifying one of eight registers
K	K	K	18-bit operand or address
x	x	x	Unused designator
A	A	A	One of eight 18-bit address registers
B	B	B	One of eight 18-bit index registers; B0 is fixed and equal to zero
X	X	X	One of eight 60-bit operand registers
( )	( )	( )	Content of a register or location
			Compare/Move
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
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

- Not applicable

CP INSTRUCTION DESCRIPTIONS

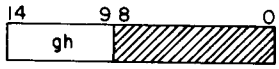
The instruction descriptions are in numerical order. The instruction shaded areas, like those in the following 00xxx and 010xK instruction formats, indicate unused bits. The unused bits are ignored by the CP.

00xxx — Error Exit to MA or Program Stop — Models 171 through 175



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 this stop occurs, the content of the P register may not correspond to the address of the 00 instruction. The P register may have been incremented prior to the execution of the 00 instruction. 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 under Central Processor Programming in section 5.)

00xxx — Error Exit to EEA — Model 176



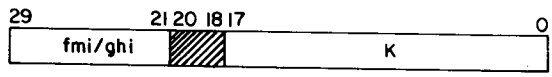
This instruction is treated as an error condition and sets the program range condition flag in the program status designator (PSD) register. This condition flag generates an error exit request which causes an exchange jump to the error exit address (EEA) register. All instructions which have issued prior to this instruction are run to completion. Any instructions following this instruction in the current instruction word (CIW) register are not executed. When all operands have arrived at the operating registers as a result of previously issued instructions, an exchange jump occurs to the exchange package designated by EEA.

The i, j, and k designators in this instruction are ignored. The program address stored in the exchange package on the terminating exchange jump advances one count from the CIW address. This is true regardless of which parcel of the CIW contains the error exit instruction.

This instruction is not intended for use in normal program code. The program range condition flag sets in the PSD register to indicate that the program has jumped to an area of central memory (CM) which may be in range but is not a valid program code. This should occur when an incorrectly coded program jumps into an unused area of CM or into a data field. The program range condition flag also sets on the condition of a jump to address zero or a jump beyond the CM field length. These conditions can be determined by the system monitor program on the basis of the register contents in the exchange package. The existence of an error exit condition resulting from execution of this instruction may thus be deduced by the monitor program.

A special situation may occur when a program is terminated with an error exit instruction, and a previously issued instruction stores a result operand in CM. The error exit is treated as a CM range error which blocks a write operation in CM as soon as the error is detected. A legitimate CM write operation may be blocked by the error condition even though the instruction causing the write issues substantially before the error exit. The timing depends upon the CM bank conflicts which may have occurred.

010xK — Return Jump to K



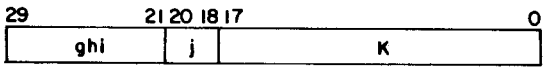
This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction writes a special word into CM at relative address K. The current program sequence then terminates by a jump to address K plus 1. The word stored in memory contains a jump instruction which causes an unconditional jump to the address of this return jump instruction plus 1. In models 175 and 176, any jump voids the instruction word stack (IWS).

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 (0400) instruction 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 — Models 171 through 175



This 30-bit instruction uses bits 0 through 17 as operand K. This 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 in A0 in CM. The length of the block of words read is the sum of the content of Bj plus K.

Three 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 field length for ECS (FLE) register in the CP are set and the content of Bj plus K is positive and nonzero, 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, stops 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 execute to completion prior to the beginning of data transfer. No further instructions issue until this block transfer is complete. 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 actual transfer of data starts.

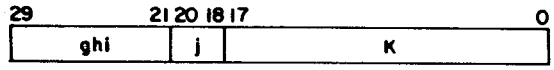
The maximum length of a block transfer is 131K 60-bit words and is determined by the addition of the signed integers in Bj and K. Both the CP and the ECS check the result of the addition, performed in an 18-bit one's-complement mode. The result is treated as an 18-bit integer. A zero or negative result 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 under Central Processor Programming in section 5.)

The normal exit for this instruction is to the content of 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.

If an exchange jump occurs during an ECS transfer, the transfer completes if only one ECS record remains. If more than one record remains, the ECS transfer terminates. In this case, the CPU P register resets so that the ECS instruction appears as if it had not issued, although some words may have transferred.

011jK Block Copy (Bj) +K Words from LCME to CM — Model 176



This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads a sequence of 60-bit words from consecutive addresses in large core memory extension (LCME) and copies them into a block of consecutive addresses in CM. The block of words begins at the LCME address formed by adding the lower 21 bits of X0 to the lower 22 bits of RAL. The lowest-order 22 bits of FLL are used for range checks. The words are stored in CM beginning at the address specified by A0. The number of words to be copied is the sum of the contents of Bj plus K. This quantity cannot exceed 177 (octal) words. If a larger quantity is used, LCME truncates the quantity to the 10-bit maximum. Thus, a block count of 3000 (octal) words transfers 1000 (octal) words. No error indications are given when this occurs unless the field length is exceeded, causing a block range error.

This instruction moves a quantity of data from LCME into CM as quickly as possible. All other activity, except input/output (I/O) word requests, stops during the block transfer of data. All instructions issued prior to this instruction execute to completion. No further instructions issue until this block transfer is nearly complete. As a result, the data flow from LCME to CM proceeds at a rate up to one 60-bit word each clock period. When an I/O word request for CM occurs during this transfer, the data flow is interrupted for 1 clock period. The I/O word address is inserted in the stream of addresses to the storage address stack (SAS), and the addresses for the block transfer resume with a minimum of a 1-clock-period delay. An additional delay occurs if the I/O reference causes a bank conflict in CM.

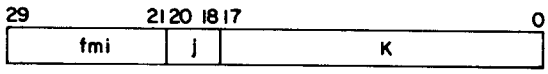
The length of the block is determined by adding K to the content of Bj. Either quantity may be used to increment or decrement the other. The addition is performed in an 18-bit one's complement mode. The result is treated as an 18-bit positive integer. This 18-bit quantity truncates to 10 bits by LCME. A zero result causes this instruction to execute as a pass instruction. The 10 bits are used for the

block count. All 18 bits are used when checking for a range error.

Three 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.

If block copy exit control (SCR bit 86) sets, this instruction exits to the next instruction in sequence. If block copy exit control clears, this instruction exits by skipping all remaining parcels in its instruction word and begins execution of the first instruction in the following word.

012jK Block Copy (Bj) + K Words from CM to ECS — Models 171 through 175



This 30-bit instruction uses bits 0 through 17 as operand K. This 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 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 and the content of Bj plus K is positive and nonzero, 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 read/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, stops 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 execute to completion prior to the beginning of data transfer. No further instructions issue until this block transfer completes. 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 starts.

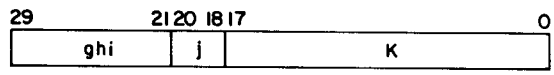
The maximum length of a block transfer is 131K 60-bit words and is determined by the addition of the signed integers in Bj and K. Both the CP and the ECS check the result of the addition, performed in an 18-bit one's-complement mode. The result is treated as an 18-bit integer. A zero or negative result 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 under Central Processor Programming 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.

If an exchange jump occurs during an ECS transfer, the transfer completes if only one ECS record remains. If more than one record remains, the ECS transfer terminates. In this case, the CPU P register resets so that the ECS instruction appears as if it had not issued, although some words may have been transferred.

012jK Block Copy (Bj) + K Words from CM to LCME — Model 176



This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads a sequence of 60-bit words from consecutive addresses in CM and copies them into a block of consecutive addresses in LCME. The block of words begins in CM at the address specified by the content of A0. The starting LCME address forms by adding the lower 21 bits of X0 the lower 22 bits of RAL. The lowest-order 22 bits of FLL are used for range checks. The number of words to be copied is the sum of the content of Bj plus K. This quantity cannot exceed 1777 (octal) words. If a larger quantity is used, LCME truncates the quantity to the 10-bit maximum. Thus, a block count of 3000 (octal) words transfers 1000 (octal) words. No error indications are given when this occurs unless the field length is exceeded, causing a block range error.

This instruction moves a quantity of data from CM into LCME as quickly as possible. All other activity, except I/O or PPS word requests, stops during the block transfer of data. All instructions issued prior to this instruction execute to completion. No further instructions issue until the block transfer is nearly complete. The rate of data flow from CM to LCME for the 012jK block copy instruction is the same as for the 011jK instruction.

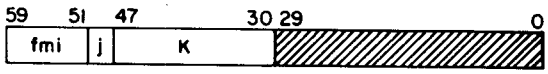
The length of the block is determined by adding K to the content of Bj. Either quantity may be used

to increment or decrement the other. The addition is performed in an 18-bit one's complement mode. The result is treated as an 18-bit positive integer. This 18-bit quantity truncates to 10 bits by LCME. A zero result causes this instruction to be executed as a pass instruction. The 10 bits are used for the block count. All 18 bits are used when checking for a range error.

Three 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.

If block copy exit control (SCR bit 86) sets, this instruction exits to the next instruction in sequence. If block copy exit control clears, this instruction exits by skipping all remaining parcels in its instruction word and begins execution of the first instruction in the following word.

013jK Central Exchange Jump to (Bj) + K (Monitor Flag Set)— Models 171 through 175



This 60-bit instruction uses bits 30 through 47 as operand K. The starting address for the exchange jump is the 18-bit result formed by adding K to the content of Bj. 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 many object program exchange packages. A selected object program exchange package then returns to this same area of CM and resumes the monitor program when its execution interval completes. (Refer to the following alternate form of the 013 instruction.)

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

The program address stored in the exchange package advances 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 under Central Processor Programming in section 5.)

013xx Central Exchange Jump to MA (Monitor Flag not Set)— Models 171 through 175

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 does not use 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 that the object program may continue at a later time. The program address in the object program exchange package advances 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.

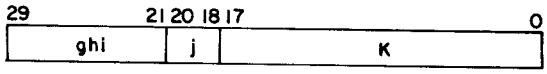
This instruction has priority over PPS exchange jump requests. If a PPS exchange jump request occurs simultaneously with the execution of this instruction, the request waits until the central exchange completes. Error exchange requests, if they occur, process before the central exchange jump executes.

The program address stored in the exchange package advances 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 under Central Processor Programming in section 5.)

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

013jK Exchange Exit to (Bj) + K (Exit Mode Flag Set) — Model 176



This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction causes the current program sequence to terminate with an exchange jump to an address in CM. The exchange package location is the relative address specified by the content of Bj plus K. The two quantities are added in an 18-bit one's complement mode. The result is treated as an 18-bit positive integer. This integer is added to the content of the reference address for CM (RAS), also treated as an 18-bit positive integer, to form the absolute address of the exchange package in CM.

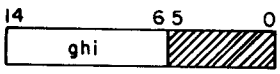
This form of the 013 instruction is used by the monitor program only. The exit mode flag in the PSD register clears during execution of object programs. The monitor program uses this instruction to exchange jump to one of many object program exchange packages. Each exchange package specifies the exit mode flag status, normally cleared. A selected object program exchange package then returns to this same area of CM and resumes the monitor program when its execution interval completes. (Refer to the alternate form of the 013 instruction.)

This instruction has priority over all other types of exchange jump requests. If an I/O interrupt request, an error exit request, or PPS request occurs prior to the execution of this instruction, the request is denied. The rejected interrupt request is not lost since the conditions which caused it are reinstated when the exchange package enters its next execution interval.

Any remaining instructions in the CIW do not execute. The program address stored in the exchange package advances one count from the address of the CIW. Therefore, the program continues at the first parcel of the following instruction word during the next execution interval for this exchange package.

The current contents of the IWS are voided by the execution of this instruction.

013xx Exchange Exit to NEA (Exit Mode Flag Not Set)— Model 176



An exchange exit instruction executed in this mode causes the current program sequence to terminate with an exchange jump to the content of the normal exit address (NEA). This is an absolute address in CM and is generally not in the CM field for the current program. This mode does not use the j or k designators in the instruction.

This instruction allows switching rapidly from an object program to a monitor program. All operating register values, program addresses, and mode selections are preserved in this process so that the object program may continue at a later time. The program address in the object program exchange package advances 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.

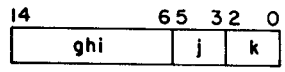
This instruction calls the system monitor program for I/O requests, library calls, storage assignments, and so on. The operating register values at the time of execution of this instruction allow parameter interchange between the object program and the monitor program.

This instruction has priority over all other types of exchange jump requests. If an I/O interrupt request or an error exit request occurs prior to the execution of this instruction, the request is denied. The rejected interrupt request is not lost since the conditions which caused it are reinstated when the exchange package enters its next execution interval.

Any remaining instructions in the CIW do not execute. The program address stored in the exchange package advances one count from the address of the CIW. Therefore, the program continues at the first parcel of the following instruction word during the next execution interval for this exchange package unless the monitor program alters the exchange package.

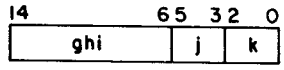
The current contents of the IWS are voided by the execution of this instruction.

014xx through 017xx Instructions — Models 171 through 175



These instructions are illegal. (Refer to Illegal Instructions under Central Processor Programming in section 5.)

014jk Read LCME at (Xk) to (Xj)— Model 176

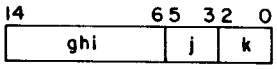


This instruction reads one word from LCME and enters the word in an X register. The word reads from LCME at the relative address specified by the lowest-order 21 bits of Xk. The word then enters the Xj register. This process does not involve CM.

This instruction is for direct addressing of individual words in LCME. The instruction may also be used for addressing a string of words in consecutive storage locations. This is advantageous if a string of words is to be read, modified, and written back into the same storage locations.

This instruction is buffered to the extent that it issues in 1 clock period unless a previous LCME reference is in process. When this instruction issues, the LCME busy flag sets and remains set until the requested word is delivered to the designated X register. This process differs from CM read reference by permitting only one LCME read or write at one time.

015jk Write Xj into LCME at (Xk)— Model 176

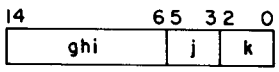


This instruction writes one word directly into LCME from an X register. The word reads from the Xj register and writes into LCME at the relative address specified by the lowest-order 21 bit of Xk. This process does not involve CM.

This instruction is for direct addressing of individual words in LCME. The instruction may also be used for addressing a string of words in consecutive storage locations. This is advantageous if a string of words is to be read, modified, and written back into the same storage locations.

This instruction is buffered to the extent that it issues in 1 clock period unless a previous LCME reference is in process. When this instruction issues, the LCME busy flag sets and remains set until the word is delivered to the proper LCME bank operand register. The following instruction may issue in the next clock period and may use either of the X registers designated in this instruction. If the word cannot be entered in the proper LCME bank operand register immediately, it is held in the LCME write register until the LCME bank operand register is free. This process differs from a CM write reference by permitting only one LCME read or write at one time.

0160k Reset Input Channel (Bk) Buffer— Model 176



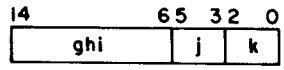
This instruction resets the input channel, specified by the content of Bk, buffer address register in preparation for the next incoming record. The input channel buffer address register clears to zero and the assembly register resets to the first position.

This instruction is for execution in the monitor program input routine which terminates a record of incoming data and prepares for the next record. The monitor input routine is called by an I/O interrupt request when the input record flag sets. The data in the buffer normally transfers to LCME by the program, and this instruction then executes to clear the buffer control for the next incoming record.

This instruction is effective only if the monitor mode flag is set in the PSD register. If the monitor mode flag is clear, this instruction becomes a pass instruction. There are no interlocks for this instruction except the monitor mode flag. When this instruction issues, it executes the required channel functions without regard to the current status or activity of the channel.

This instruction is normally executed by the program in response to an I/O interrupt request resulting from the setting of the input record flag. This flag clears when the interrupt request occurs. Further entries to the buffer are not locked out by the interrupt request flag in the channel access control during the execution interval for the interrupt exchange package. The equipment connected to the input channel must wait for a positive response from the monitor program over the output channel before beginning the next record.

016jk Read Input Channel (Bk) Status to Bj (j≠0) — Model 176



This instruction reads the current value of the input channel, specified by the content of Bk buffer address register to Bj. The status of the buffer address register is not altered.

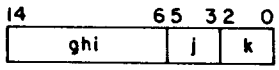
This instruction monitors the progress of the input channel buffer. The buffer area is divided into two fields by the threshold testing mechanism. Each half of the buffer area constitutes one field. An I/O interrupt request is generated by the threshold testing mechanism whenever the input channel buffer address advances across a field boundary. This occurs at the center and at the end of the buffer area.

This instruction allows a monitor program to determine whether an I/O interrupt request was generated by a buffer threshold test or by a record flag. The monitor program must retain the buffer address from one interrupt period to the next. If the buffer address is in the same field as the previous interrupt, the interrupt request was from a record

flag. If the buffer address is in the opposite field from the previous interrupt, the interrupt request was from a theshold test.

If the Bk channel number is zero, the current content of the CPU clock period counter reads into Bj. This is a 17-bit counter which advances one count in a two's complement mode each clock period. This count is for timing measurements of programs. Timing considerations for this special use are the same as the normal timing for an input channel buffer address register.

0170k Reset Output Channel (Bk) Buffer— Model 176

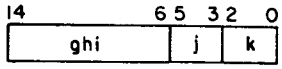


This instruction resets the output channel, specified by the content of Bk, buffer address register in preparation for the next record transmission. In a normal-speed channel, the output channel buffer address register clears to zero until the equipment connected to the channel accepts the first word. In a high-speed channel, the buffer address register contains a count of one before the first word is accepted. A record pulse is transmitted on the output channel data path. The output word request flag then sets to permit a read of the first word from the buffer.

This instruction is for execution in the monitor program output routine to initiate a new record transmission over a channel output data path. The buffer is normally inactive when this instruction executes. The buffer loads with the data for the next record, and this instruction then executes to initiate the transmission. A record pulse is transmitted to indicate the beginning of a new record. The first word of data follows as soon as the output word request flag has caused the first word to be read from the output buffer to the disassembly register.

This instruction is effective only if the monitor mode flag is set in the PSD register. If the monitor mode flag is clear, this instruction becomes a pass instruction. There are no interlocks for this instruction except the monitor mode flag. When this instruction issues, it executes the required channel functions without regard to the current status or activity of the channel. The disassembly register is reset by the output word request flag.

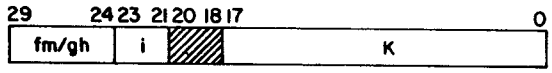
017jk Read Output Channel (Bk) Status to Bj (j≠0)— Model 176



This instruction reads the current value of the output channel, specified by the content of Bk, buffer address register to Bj. The status of the buffer address register is not altered.

This instruction monitors the progress of the output channel buffer. The buffer area is divided into two fields by the threshold testing mechanism. Each half of the buffer area constitutes one field. An I/O interrupt request is generated by the threshold testing mechanism whenever the buffer address advances across a field boundary. This occurs at the center of the buffer area and at the end of the buffer area.

02ixK Jump to (Bi)+ K

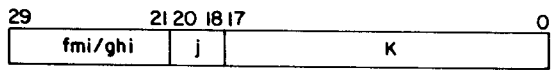


This two-parcel instruction uses the lower-order 18 bits as operand K. In model 175, this instruction unconditionally voids the IWS. The instruction causes the current program sequence to terminate with a jump to address Bi plus K in CM.

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 one's-complement mode. The result is treated as an 18-bit positive integer which 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



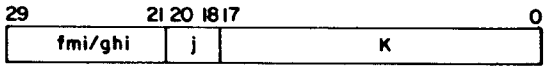
This two-parcel instruction uses the lower-order 18 bits as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of Xj. The branch to address K occurs only on the following conditions. The current program sequence continues 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 branches on a zero result from either a fixed-point or a floating-point operation.

In model 175, a jump from the IWS voids the stack.

031jK Branch to K if (Xj)≠0



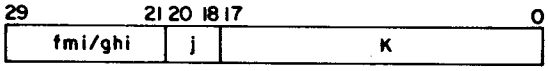
This two-parcel instruction uses the lower-order 18 bits as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon the content of Xj. The program sequence continues only on the following 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 branches on a nonzero result from either a fixed-point or a floating-point operation.

In model 175, a jump from the IWS voids the stack.

032jK Branch to K if (Xj) Positive



This two-parcel instruction uses the lower-order 18 bits 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 Xj. The branch decision for this instruction is based on the value of the sign bit in Xj.

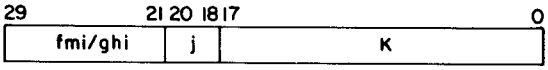
Jump to K if: Bit 59 of Xj = 0 (positive)

Continue if: Bit 59 of Xj = 1 (negative)

This instruction branches on a positive result from either a fixed-point or a floating-point operation.

In model 175, a jump from the IWS voids the stack.

033jK Branch to K if (Xj) Negative



This two-parcel instruction uses the lower-order 18 bits 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 Xj. The branch decision for this instruction is based on the value of the sign bit in Xj.

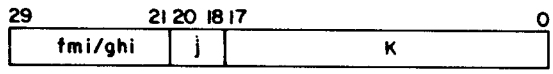
Jump to K if: Bit 59 of Xj = 1 (negative)

Continue if: Bit 59 of Xj = 0 (positive)

This instruction branches on a negative result from either a fixed-point or a floating-point operation.

In model 175, a jump from the IWS voids the stack.

034jK Branch to K if (Xj) in Range



This two-parcel instruction uses the lower-order 18 bits 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 Xj. The program sequence continues only on the following conditions. The branch to address K occurs for all other cases.

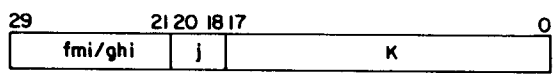
Continue if:

(Xj) = 3777 xxxx xxxx xxxx xxxx (positive overflow)	Models 171 through 176
(Xj) = 4000 xxxx xxxx xxxx xxxx (negative overflow)	
(Xj) = 1777 xxxx xxxx xxxx xxxx (positive indefinite)	Model 176
(Xj) = 6000 xxxx xxxx xxxx xxxx (negative indefinite)	

This instruction branches 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 test.

In model 175, a jump from the IWS voids the stack.

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



This two-parcel instruction uses the lower-order 18 bits 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 Xj. The branch to address K occurs only on the following conditions. The current program sequence continues for all other cases.

Jump to K if:

(Xj) = 3777 xxxx xxxx xxxx xxxx (positive overflow)
(Xj) = 4000 xxxx xxxx xxxx xxxx (negative overflow)
} Models 171 through 176

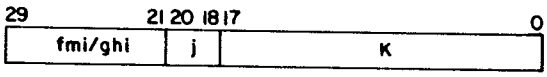
(Xj) = 1777 xxxx xxxx xxxx xxxx (positive indefinite)
(Xj) = 6000 xxxx xxxx xxxx xxxx (negative indefinite)
} Model 176

This instruction branches 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 test.

In models 171 through 175, overflow is not in range. In model 176, overflow and indefinite are not in range.

In model 175, a jump from the IWS voids the stack.

036jK Branch to K if (Xj) Definite



This two-parcel instruction uses the lower-order 18 bits 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 Xj. The program sequence continues only on the following conditions. The branch to address K occurs for all other cases.

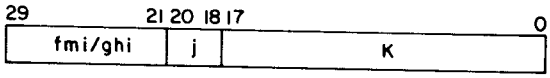
Continue if:

(Xj) = 1777 xxxx xxxx xxxx xxxx (positive indefinite)
(Xj) = 6000 xxxx xxxx xxxx xxxx (negative indefinite)

This instruction branches 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 test.

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

037jK Branch to K if (Xj) Indefinite



This two-parcel instruction uses the lower-order 18 bits 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 continues for all other cases.

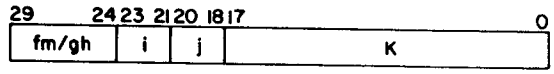
Jump to K if:

(Xj) = 1777 xxxx xxxx xxxx xxxx (positive indefinite)
(Xj) = 6000 xxxx xxxx xxxx xxxx (negative indefinite)

This instruction branches 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 test.

In model 175, a jump from the IWS voids the stack.

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

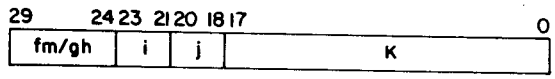


This two-parcel instruction uses the lower-order 18 bits as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of the Bi and Bj registers. The branch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

This instruction branches 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 from the IWS voids the stack.

05ijK Branch to K if (Bi) ≠ (Bj)

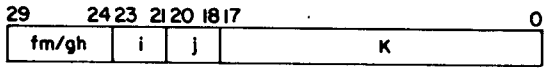


This two-parcel instruction uses the lower-order 18 bits as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of the Bi and Bj registers. The program sequence continues 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 branches 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 from the IWS voids the stack.

06ijK Branch to K if (Bi) ≥ (Bj)

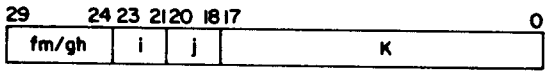


This two-parcel instruction uses the lower-order 18 bits as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of Bi and Bj. Both quantities are 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 continues if the content of Bi is less than Bj.

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

In model 175, a jump from the IWS voids the stack.

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

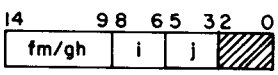


This two-parcel instruction uses the lower-order 18 bits as operand K. Execution of this instruction causes the program sequence to terminate with a jump to address K in CM or to continue with the current program sequence, depending upon a comparison of the contents of Bi and Bj. Both quantities are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

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

In model 175, a jump from the IWS voids the stack.

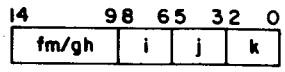
10ijx Transmit (Xj) to Xi



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

This instruction moves 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

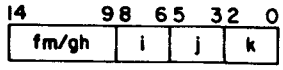


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The result delivered to Xi is the bit-by-bit logical product of the two operands. Each of the 60 bits in Xj is 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 extracts portions of a 60-bit word during data processing.

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

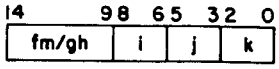


This instruction reads operands from two X registers, operates upon them to form a result, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. The result delivered to Xi is the bit-by-bit logical sum of the two operands. Each of the 60 bits in Xj is compared 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 merges 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 result delivered to Xi 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.

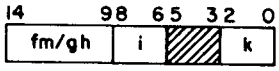
(Xj) = 0123 7777 0123 4567 1010

(Xk) = 0123 4567 7777 3210 1100

(Xi) = 0000 3210 7654 7777 0110

This instruction compares bit patterns of complements bit patterns during data processing.

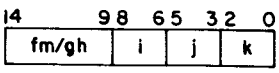
14ixk Transmit Complement of (Xk) to Xi



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

This instruction changes the sign of a fixed-point or floating-point quantity. The instruction also inverts 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 result delivered to Xi 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.

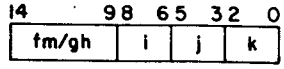
(Xj) = 7777 7000 0123 4567 1010

(Xk) = 0123 4567 0007 7700 1100

(Xi) = 7654 3000 0120 0067 0010

This instruction extracts 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 result delivered to Xi 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.

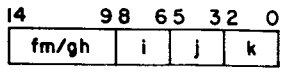
(Xj) = 0000 7777 0123 4567 1010

(Xk) = 0123 4567 7777 0000 1100

(Xi) = 7654 7777 0123 7777 7677

This instruction merges 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 result delivered to Xi 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.

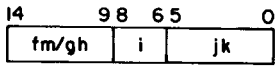
(Xj) = 0123 7777 0123 4567 1010

(Xk) = 0123 4567 7777 3210 1100

(Xi) = 7777 4567 0123 0000 7667

This instruction compares bit patterns or complements bit patterns during data processing.

20ijk Left Shift (Xi) by jk



This instruction reads one operand from Xi, 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 j and k designators are treated as a single 6-bit positive integer operand in this instruction.

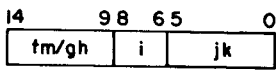
A left circular shift implies that the bit pattern in the 60-bit word is displaced towards the highest-order bit positions. The bits shifted off the upper end of the 60-bit word are inserted in the lowest-order bit positions in the same sequence. The resulting 60-bit word has the same quantity of bits 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 Xi, 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 j and k designators are treated as a single 6-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 shifted off the lower end of the word are discarded. The highest-order bit positions are filled with copies of the original sign bit.

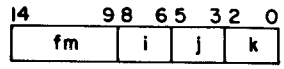
Two sample computations are listed in octal notation to illustrate the operation performed. The first example contains a positive operand, and the second example contains a negative operand.

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 — Models 171 through 175



This instruction reads a 60-bit operand from Xk, shifts the data either left or right as specified by Bj, and writes the resulting 60-bit word into Xi. If the value in Bj is positive, the data is left shifted circularly the number of bit positions designated by the value in Bj. If the value in Bj is negative, the data is right shifted 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 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 positions. The bits 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, and the second example illustrates the right shift with sign extension.

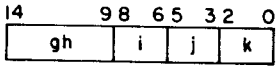
(Xk) = 2323 6600 0000 0000 0111
(Bj) = 00 0012
(Xi) = 7540 0000 0000 0022 2464

(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 and Bj bit 17 is set, the shift count is greater than 63 (decimal) places right, and a result of positive zero is returned to Xk. Bj bits 11 through 16 are not tested by this instruction.

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

22ijk Left Shift (Xk) Nominally (Bj) Places to Xi— Model 176



This instruction reads a 60-bit operand from Xk, shifts the data either left or right as specified by the content of Bj, and writes the resulting 60-bit word into Xi. If the value in Bj is positive, the data is left shifted circularly the number of bit positions designated by the value in Bj. If the value in Bj is negative, the data is right shifted 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 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 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, and 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 11 are different than Bj bit 17 and Bj bit 17 is set, the shift count is greater than 63 (decimal) places right, and a result of negative zero is returned to Xk. Bj bits 12 through 16 are not tested by this instruction.

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

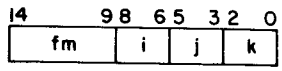
If Bj is zero (000000 or 777777), this instruction reads the operand from Xk and copies it unaltered into Xi. The timing is the same as for the normal case.

If Bj is positive, only the lowest-order six bits are used in determining the shift count. The highest-order bits are ignored. The resulting 6-bit shift count is treated modulo 60 (decimal). For example, a shift count of 63 (decimal) results in a left circular shift of three bit positions.

If Bj is negative, only the lowest-order 12 bits are used in determining the shift count. The highest-order bits are ignored. The lowest-order 12 bits of Bj are complemented, and the resulting positive integer determines the shift count. If this shift count is greater than 60 (decimal), the result stored in Xi consists of 60 copies of the original operand sign bit.

An all ones or all zeros word is treated in the same manner as any other bit pattern. The timing is the same as for the normal case.

23ijk Right Shift (Xk) Nominally (Bj) Places to Xi — Models 171 through 175



This instruction reads a 60-bit operand from Xk, shifts the data either left or right as specified by the content of Bj, and writes the resulting 60-bit word into Xi. If the value in Bj is positive, the data is right shifted with sign extension the number of bit positions designated by the value in Bj. If the value in Bj is negative, the data is left shifted circularly 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 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 words is displaced towards the lowest-order bit positions. The bits 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 resulting in a right shift with sign extension, and the second example contains a negative shift count resulting 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

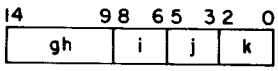
(Bj) = 77 7765

(Xi) = 7540 0000 0000 0022 2464

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

This instruction is used when the amount of shift is derived in the computation. The instruction is also used 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— Model 176



This instruction reads a 60-bit operand from Xk, shifts the data either left or right as specified by the content of Bj, and writes the resulting 60-bit word into Xi. If the value in Bj is positive, the data is right shifted with sign extension the number of bit positions designated by the value in Bj. If the value in Bj is negative, the data is left shifted circularly 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 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 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 resulting in a right shift with sign extension, and the second example contains a negative shift count resulting 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 11 are different than Bj bit 17 and Bj bit 17 is set, the shift count is greater than 63 (decimal) places right, and a result of negative zero is returned to Xk. Bj bits 12 through 16 are not tested by this instruction.

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

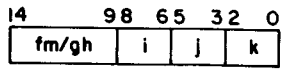
If Bj is zero (000000 or 777777), this instruction reads the operand from Xk and copies it unaltered into Xi. The timing is the same as for the normal case.

If Bj is positive, only the lowest-order 12 bits are used in determining the shift count. The highest-order bits are ignored. If this resulting 12-bit shift count is greater than 60 (decimal), the result stored in Xi consists of 60 copies of the original operand sign bit.

If Bj is negative, only the lowest-order six bits are used in determining the shift count. The highest-order bits are ignored. The lowest-order six bits of the content of Bj are complemented, and the resulting positive integer shift count is treated modulo 60 (decimal). For example, a shift count of 63 (decimal) results in a left circular shift of three bit positions.

An all ones or all zeros word is treated in the same manner as any other bit pattern. The timing is the same as for the normal case.

24ijk Normalize (Xk) to Xi and Bj



This instruction reads one operand from Xk, performs a normalizing operation on this word in a floating-point format, and delivers the normalized result to Xi. In addition, a positive integer shift count is sent to Bj. 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 result 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 position. The exponent is then decreased by the number of bit positions shifted.

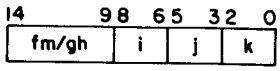
Two sample computations are listed in octal notation to illustrate the operation performed. The first example involves a positive floating-point number, and the second example involves a negative number.

(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 copies to Xi, and Bj is set to zero. In models 171 through 175, corresponding infinite and indefinite exit conditions are also set in the CP for exit mode action. In model 176, no condition flags are set in the PSD register.

25ijk Round Normalize (Xk) to Xi and Bj



This instruction reads one operand from Xk, performs a rounding and then a normalizing operation in floating-point format, and delivers the round normalized result to Xi. In addition, a positive integer shift count is sent to Bj. 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 increases 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.

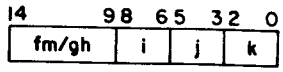
Two sample computations are listed in octal notation to illustrate the normalizing operation performed. The first example involves a positive floating-point number, and the second example involves a negative number.

(Xk) = 2034 0047 6500 0000 2262
(Xi) = 2026 4765 0000 0022 6420
(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 copies to Xi, and Bj is set to zero. In models 171 through 175, corresponding infinite and indefinite conditions are also set in the CP for exit mode action. In model 176, no condition flags are set in the PSD register.

26ijk Unpack (Xk) to Xi and Bj



This instruction reads one operand from Xk, unpacks this word from floating-point format, and delivers the coefficient and exponents to Xi and Bj, respectively. The 60-bit word delivered to Xi 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 Bj 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

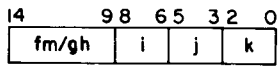
(Xk) = 1743 4500 3333 2000 0077
(Xi) = 0000 4500 3333 2000 0077
(Bj) = 77 7743

(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 converts 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 Xi. 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

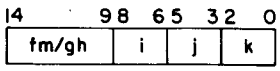
(Xk) = 0000 4500 3333 2000 0077
(Bj) = 77 7743
(Xi) = 1743 4500 3333 2000 0077

(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 converts 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 Xi in floating-point format and is not necessarily normalized.

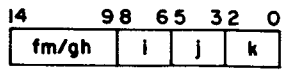
The two operands are unpacked from floating-point format, and the exponents are compared. The coefficient with the smaller exponent is right shifted by the difference of the two exponents such that both coefficients are the same significance. The two coefficients are then added to form a 96-bit result. The upper half of the 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.

When the difference between the exponents is greater than 128 (decimal), models 171 through 175 extend the shifted sign bit to the entire shifted operand. Model 176 enters a shifted operand of plus 0 regardless of the sign of the shifted operand. If the reference operand has a zero coefficient, the results can differ in sign.

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in the PSD register for the corresponding conditions. (Refer to Processing Differences under Central Processor Programming 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 Xi in floating-point format and is not necessarily normalized.

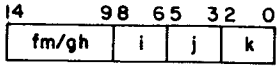
The two operands are unpacked from floating-point format, and the exponents are compared. The coefficient with the smaller exponent is right shifted by the difference of the two exponents such that both coefficients are the same significance. The Xk coefficient is then subtracted from the Xj coefficient 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.

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in the PSD register for the Corresponding Conditions. (Refer to Processing Differences under Central Processor Programming in section 5.)

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

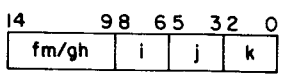


This instruction reads operands from two X registers, operates upon them to form a double-precision floating-point sum, and delivers the lower half of this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The sum of the quantities in Xj and Xk is delivered to Xi 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 smaller exponent is right shifted by the difference of the two exponents such that both coefficients are the same significance. The two 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.

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in the PSD register for the Corresponding Conditions. (Refer to Processing Differences under Central Processor Programming in section 5.)

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

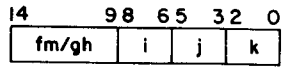


This instruction reads operands from two X registers, operates upon them to form a double-precision floating-point difference, and delivers the lower half of this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not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

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in the PSD register for the Corresponding Conditions. (Refer to Processing Differences under Central Processor Programming in section 5.)

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

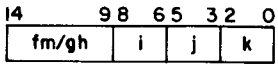


This instruction reads operands from two X registers, operates upon them to form a rounded floating-point sum, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result is delivered to Xi 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. A round bit is always inserted in the coefficient with the larger exponent. If the two 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 are normalized or have unlike signs. The second round bit is inserted before the coefficient has been shifted by the difference of the exponents.

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in the PSD register for the Corresponding Conditions. (Refer to Processing Differences under Central Processor Programming in section 5.)

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

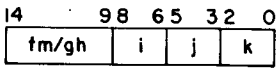


This instruction reads operands from two X registers, operates upon them to form a rounded floating-point difference, and delivers this result to a third X register. The operands for this instruction are in Xj and Xk. These operands are in floating-point format and are not necessarily normalized. The result of subtracting the quantity in Xk from the quantity in Xj is delivered to Xi 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. A round bit is always inserted in the coefficient with the larger exponent. If the two 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 have unlike signs. The second round bit is inserted before the coefficient has been shifted by the difference of the exponents.

For models 171 through 175, infinite (3777xxx...x or 4000xxx..,x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in the PSD register for the corresponding conditions. (Refer to Processing Differences under Central Processor Programming in section 5.)

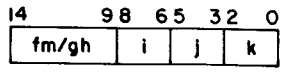
36ijk Integer Sum of (Xj) and (Xk) to Xi



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 Xi. Overflow is not detected.

This instruction adds integers too large for handling by 50 through 77 instructions. The instruction also merges and compares data fields during data processing.

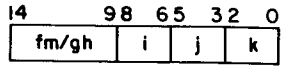
37ijk Integer Difference of (Xj) and (Xk) to Xi



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 Xi. Overflow is not detected.

This instruction subtracts integers too large for handling by 50 through 77 instructions. The instruction also compares 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 Xi 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 1-bit left shift to normalize the result coefficient is done. The resulting exponent is reduced by one count in this case.

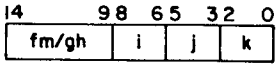
If both operands are not normalized, the resulting double-precision product has less than 96 significant bits. No test is made for the position of the most significant bit. The upper 48 bits are read from the double-precision product register. Leading zeros occur in this result coefficient.

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

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action.

For model 176, bits are set in the PSD register for the corresponding conditions. (Refer to Processing Differences under Central Processor Programming 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 Xi 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.

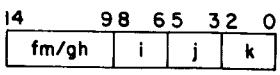
The two operands are unpacked from floating-point format. The exponents are added with a correction factor to determine the exponent for the result. The coefficients are multiplied as signed integers to form a 96-bit integer product. A rounding bit is added to bit position 46 of this product. The upper 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 1-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. The upper 48 bits are read from the double-precision product register. Leading zeros occur in this result coefficient.

This instruction is used in single-precision floating-point calculations. For multiple-precision calculations, the 40 and 42 instructions must be used.

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in PSD register for the corresponding conditions. (Refer to Processing Differences under Central Processor Programming 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 Xi in floating-point format and is not necessarily normalized.

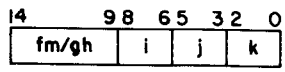
The operands are not rounded in this operation. The two operands are unpacked from floating-point format. The exponents are added to determine the exponent for the result. The result exponent is exactly 48 less than the exponent for a 40 instruction. The coefficients are multiplied as signed integers 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 1-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. The lower 48 bits are always read from the 96-bit product register.

This instruction is used 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 Xi 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. If the result is nonzero, overflow is then indicated. An integer multiply operation is not intended to be used with normalized operands.

For models 171 through 175, infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands cause corresponding exit conditions to set in the CP for exit mode action. For model 176, bits are set in the PSD register for the corresponding conditions. (Refer to Processing Differences under Central Processor Programming 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 6-bit octal 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 Xi. The following are sample parameters.

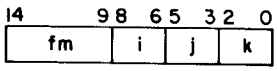
j = 2

k = 4

Xi = 7777 7760 0000 0000 0000

This instruction generates 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 — Models 171 through 175



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 Xi. 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 are assembled, they are packed with the result exponent into floating-point format and delivered to Xi.

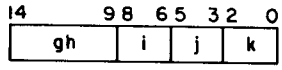
If the exponent subtraction causes an underflow or overflow, an underflow or overflow result is returned even with the occurrence of a divide fault.

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 Processing Differences under Central Processor Programming in section 5.)

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 content of Xj is larger than the coefficient for the content of 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 under Central Processor Programming in section 5.)

This instruction is used 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.

44ijk Floating Divide (Xj) by (Xk) to Xi— Model 176



This instruction causes the divide unit to read operands from two X registers, operate upon them to form a floating-point quotient, and deliver this result to a third X register. The operands for this instruction are the contents of 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 Xi. If both operands are normalized, the quotient is also normalized. The remainder from the division process is discarded.

The two operands are not rounded in this operation. The 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 the content of Xj is positioned in a dividend register. The coefficient from the content of Xk is trial-subtracted repeatedly from the dividend, and the dividend is shifted to form the quotient bits. The quotient bits are assembled in a quotient register. When 48 bits of the quotient are assembled, they are packed with the result exponent into floating-point format and delivered to Xi.

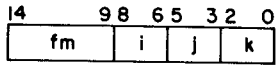
If the exponent subtraction causes an underflow or overflow, an underflow or overflow result is returned even with the occurrence of a divide fault.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, bits are set in the PSD register for the corresponding conditions. (Refer to Processing Differences under Central Processor Programming in section 5.)

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 content of Xj is larger than the coefficient for the content of Xk by a factor of two or more, the quotient is incorrect.

This instruction is used 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 divident in order to reconstruct the remainder.

45ijk Round Floating Divide (Xj) by (Xk) to Xi — Models 171 through 175



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 Xi. If both operands are normalized, the quotient is also normalized. The remainder from the division process is discarded.

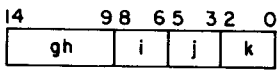
The two operands are unpacked from floating-point format in this operation. The exponents are subtracted with a correction factor to determine the exponent for the result. The coefficient from Xj is positioned in a dividend register. The Xj quantity is modified by inserting a 2525...25 round pattern 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 are assembled, they are packed with the result exponent into floating-point format and delivered to Xi.

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 under Central Processor Programming in section 5.)

This instruction is used 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.

45ijk Round Floating Divide (Xj) by (Xk) to Xi—Model 176



This instruction causes the divide unit to read operands from two X registers, operate upon them to form a rounded floating-point quotient, and deliver this result to a third X register. The operands for this instruction are in the contents of 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 Xi. 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 the content of Xj is positioned in a dividend register. The Xj quantity is modified by adding a round bit below the lowest-order bit of the coefficient from the content of Xj. This round bit increases the magnitude of the dividend by one-half the value of the least significant bit. The coefficient from the content of Xk is trial-subtracted repeatedly from the dividend, and the dividend is shifted to form the quotient bits. The quotient bits are assembled in a quotient register. When 48 bits of the quotient are assembled, they are packed with the result exponent into floating-point format and delivered to Xi.

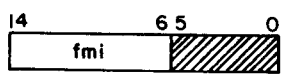
If the dividend is not normalized, the quotient cannot be normalized. However, the quotient is correct even though there may be leading zeros in the coefficient. If the divisor is not normalized, the quotient may be incorrect. If the coefficient for the content of Xj is larger than the coefficient for the content of Xk by a factor of two or more, the quotient is incorrect.

This instruction is used in single-precision floating-point calculations where rounding of operands is desired to reduce truncation errors.

The rounding step occurs in the dividend register prior to the first trial subtraction. A round bit is added to the dividend which has the effect of increasing the dividend by one-half the value of the least significant bit. The effect on the quotient varies depending upon the value of the divisor and upon the truncation point in the quotient. If the dividend is smaller than the divisor, the quotient is truncated one bit position lower than if the dividend is equal to or larger than the divisor. These effects cause the rounding to vary in the quotient from one-fourth the value of the least significant bit in the result to almost one.

If infinite (3777xxx...x or 4000xxx...x) or indefinite (1777xxx...x or 6000xxx...x) operands are used, bits are set in the PSD register for the corresponding conditions. (Refer to Processing Differences under Central Processor Programming in section 5.)

460xx through 463xx Pass— Models 171 through 175



These instructions fill program instruction words where necessary to match jump destinations with word boundaries. The j and k designators are

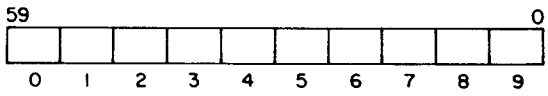
ignored, and a nonzero value has no effect in this instruction.

464 through 467 Compare/Move — Models 171 through 174

These instructions apply to model 171 only if it has the optional compare/move unit.

These instructions must appear in parcel 0 or be treated as illegal instructions. For model 175, these instructions are illegal. (Refer to Illegal Instructions under Central Processor Programming in section 5.)

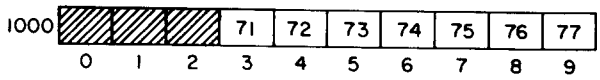
Data fields consisting of 6-bit characters may start or end with any character position (offset) of the ten 6-bit positions in each word. The character positions are designated as follows:



For move instructions, the K1 designator specifies which CM word contains the first character of the source data field, and the C1 designator specifies the character position (offset) of the first character. The K2 designator specifies the CM location in which the first character of the result data field is placed, and the C2 designator 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.



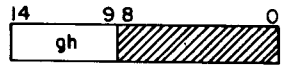
Therefore, the first character of the source field is 71.

An address is out-of-range if C1 or C2 is greater than 9, K1 plus N1 is greater than the program field length for CM (FLC), or K2 plus N2 is greater than FLC. N1 equals the number of CM references made to the source data field starting at K1, and N2 equals the 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 127 (177_8) characters. The maximum data field length for the move indirect instruction is 8191 (17777_8) 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 (octal) characters, because an exchange jump interrupts any compare/move operation having a decremented field length greater than 377 (octal).

46xxx Pass — Model 176



This instruction causes no action in any functional unit. It is used to fill program instruction words where necessary to match jump destinations with word boundaries. The i, j, and k designators are normally zero in this instruction. However, these designators are ignored, and a nonzero value has no effect.

464jK Move Indirect — Models 171 through 174

This instruction applies to model 171 only if it has the optional compare/move unit.

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.

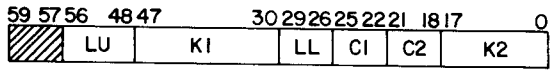


Figure 4-2. Descriptor Word Format

465 Move Direct — Models 171 through 174

This instruction applies to model 171 only if it has the optional compare/move unit.

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 7-bit count.

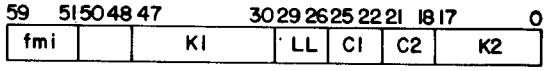


Figure 4-3. Compare/Move Instruction Format

466 Compare Collated — Models 171 through 174

This instruction applies to model 171 only if it has the optional compare/move unit.

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 and looked up in the collating table (table 4-2) beginning at address A0. If the table values found for the two unequal characters are equal, 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 (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

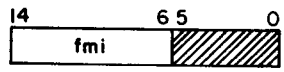
59 0

467 Compare Uncollated — Models 171 through 174

This instruction applies to model 171 only if it has the optional compare/move unit.

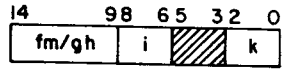
This instruction is similar to the 466 instruction except that the collating table is not used. The X0 register is set when the first pair of unequal characters is encountered or when the field length is exhausted.

464 through 467 Instructions — Model 175



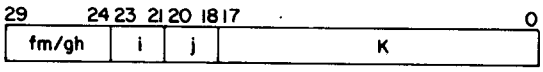
These instructions are illegal instructions. (Refer to Illegal Instructions under Central Processor Programming in section 5.)

47ixk Population Count of (Xk) to Xi



This instruction reads one operand from Xk, counts the number of one bits in the operand, and stores the count in Xi. The count delivered to Xi is a positive integer. If the operand is all ones, a count of 60 (decimal) is delivered to Xi. If the operand is all zeros, a zero word is delivered to Xi.

50ijK Set Ai to (Aj) + K

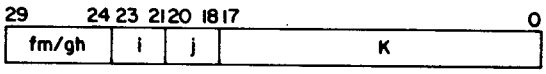


This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Aj, forms the sum of the operand plus K, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference 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 obtains operands from CM for computation and delivers the result back into CM.

51ijK Set Ai to (Bj) + K

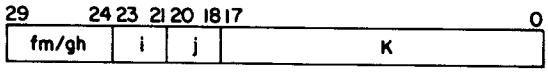


This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Bj, forms the sum of the operand plus K, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM refernce
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

This instruction obtains operands from CM for computation and delivers the result back into CM.

52ijK Set Ai to (Xj) + K

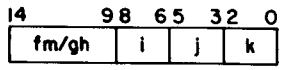


This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Xj, forms the sum of the operand plus K, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference 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)

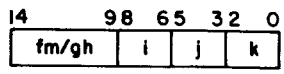


This instruction reads operands from Xj and Bk, forms the sum of the operands, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

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 obtains operands from CM for computation and delivers the result back into CM.

54ijk Set Ai to (Aj) + (Bk)

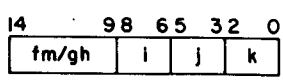


This instruction reads operands from Aj and Bk, forms the sum of the operands, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

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 obtains operands from CM for computation and delivers the result back into CM.

55ijk Set Ai to (Aj) - (Bk)

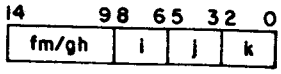


This instruction reads operands from Aj and Bk, subtracts the Bk operand from the Aj operand, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

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 obtains operands from CM for computation and delivers the results back into CM.

56ijk Set Ai to (Bj) + (Bk)



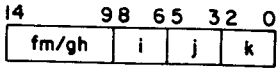
This instruction reads operands from Bj and Bk, forms the sum of the operands, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

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 obtains operands from CM for computation and delivers the results back into CM.

57ijk Set Ai to (Bj) - (Bk)

57ijk Set Ai to (Bj) - (Bk)

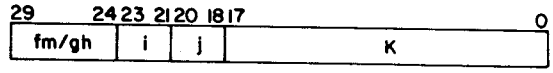


This instruction reads operands from Bj and Bk, subtracts the Bk operand from the Bj operand, and delivers the result to Ai. If the i designator is nonzero, a reference is made to CM using the result as the relative address. The type of reference is a function of the i designator value.

i = 0	No CM reference
i = 1, 2, 3, 4, 5	Read from CM to Xi
i = 6, 7	Write into CM from Xi

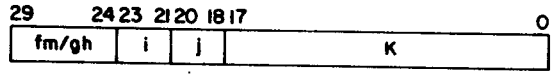
This instruction obtains operands from CM for computation and delivers the result back into CM.

60ijK Set Bi to (Aj) + K



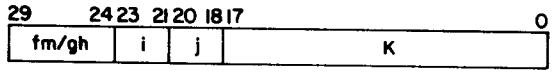
This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand plus K, and delivers the result to Bi. The sum is formed in an 18-bit one's-complement mode. This instruction is for address modification in the increment registers.

61ijK Set Bi to (Bj) + K



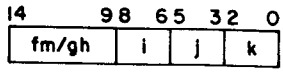
This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Bj, forms the sum of the operand plus K, and delivers the result to Bi. The sum is formed in an 18-bit one's-complement mode.

62ijK Set Bi to (Xj) + K



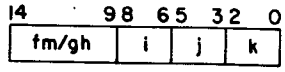
This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Xj, forms the sum of the operand plus K, and delivers the result to Bi. The sum is formed in an 18-bit one's-complement mode.

63ijk Set Bi to (Xj) + (Bk)



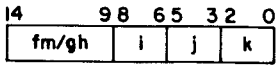
This instruction reads operands from Xj and Bk, adds the operands, and delivers the result to Bi. The sum is formed in an 18-bit one's-complement mode.

64ijk Set Bi to (Aj) + (Bk)



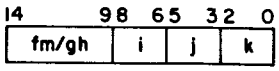
This instruction reads operands from Aj and Bk, adds the operands, and delivers the result to Bi. The sum is formed in an 18-bit one's-complement mode.

65ijk Set Bi to (Aj) - (Bk)



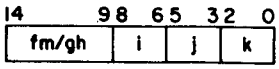
This instruction reads operands from Aj and Bk, subtracts the Bk operand from the Aj operand, and delivers the result to Bi. The difference is formed in an 18-bit one's-complement mode.

66ijk Set Bi to (Bj) + (Bk)



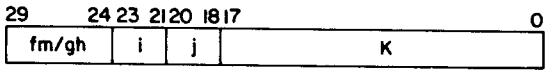
This instruction reads operands from Bj and Bk, adds the operands, and delivers the result to Bi. The sum is formed in an 18-bit one's-complement mode.

67ijk Set Bi to (Bj) - (Bk)



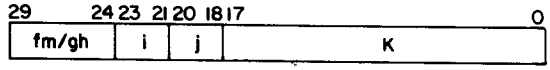
This instruction reads operands from Bj and Bk, subtracts the Bk operand from the Bj operand, and delivers the result to Bi. The difference is formed in an 18-bit one's-complement mode.

70ijK Set Xi to (Aj) + K



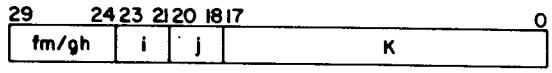
This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Aj, forms the sum of the operand plus K, and delivers the result to Xi. The sum is formed in an 18-bit one's-complement mode. The 18-bit result is sign-extended by copying the highest-order bit or the result into the upper 42 bit positions in Xi.

71ijK Set Xi to (Bj) + K



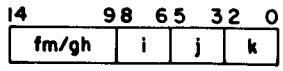
This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Bj, forms the sum of the operand plus K, and delivers the result to Xi. The sum is formed in an 18-bit one's-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in Xi.

72ijK Set Xi to (Xj) + K



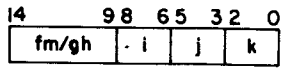
This two-parcel instruction uses the lower-order 18 bits as operand K. This instruction reads an operand from Xj, forms the sum of the operand plus K, and delivers the result to Xi. The sum is formed in an 18-bit one's-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in Xi.

73ijk R Set Xi to (Xj) + (Bk)



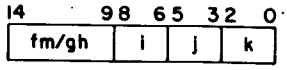
This instruction reads operands from Xj and Bk, adds the operands, and delivers the result to Xi. The sum is formed in an 18-bit one's-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in Xi.

74ijk Set Xi to (Aj) + (Bk)



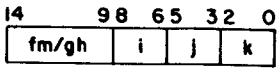
This instruction reads operands from Aj and Bk, adds the operands, and delivers the result to Xi. The sum is formed in an 18-bit one's-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in Xi.

75ijk Set Xi to (Aj) - (Bk)



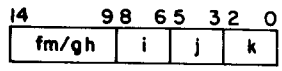
This instruction reads operands from Aj and Bk, subtracts the Bk operand from the Aj operand, and delivers the result to Xi. The difference is formed in an 18-bit one's-complement mode. The 18-bit result is sign-extended by copying the highest-order bit of the result into the upper 42 bit positions in Xi.

76ijk Set Xi to (Bj) + (Bk)



This instruction reads operands from Bj and Bk, adds the operands, and delivers the result to Xi. The sum is formed in an 18-bit one's-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 Xi.

77ijk Set Xi to (Bj) - (Bk)



This instruction reads operands from Bj and Bk, subtracts the Bk operand from the Bj operand, and delivers the result to Xi. The difference is formed in an 18-bit one's-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 Xi.

CP INSTRUCTION TIMING — MODELS 171 THROUGH 174

Execution times for the CP instructions are listed in table 4-3 for models 171 through 174. The execution times are listed with the assumption that no conflicts occur. Execution delays result unless all the conditions listed in the timing notes columns exist for the particular instruction. The numbers in the timing notes column refer to notes listed at the end of the table. Execution times are in clock periods, which are each 50 nanoseconds.

TABLE 4-3. CP INSTRUCTION TIMING - MODELS 171 THROUGH 174

Instruction Code	Description	Execution Time (Clock Periods)			Timing Notes
		Model 171	Model 172	Models 173/174	
00xxx	Error exit to MA or program stop	-	-	-	10
010xK	Return jump to K	43	36	29	-
011jK	Block copy (Bj) + K words from ECS to CM	-	-	-	4
012jK	Block copy (Bj) + K words from CM to ECS	-	-	-	4
013jK	Central exchange jump to (Bj) + K (monitor flag set)	56	49	42	-
013xx	Central exchange jump to MA (monitor flag not set)				
02ixK	Jump to (Bi) + K	36	29	22	-
030jK	Branch to K if (Xj) = 0	36	29	22	2
031jK	Branch to K if (Xj) ≠ 0	36	29	22	2
032jK	Branch to K if (Xj) positive	36	29	22	2
033jK	Branch to K if (Xj) negative	36	29	22	2
034jK	Branch to K if (Xj) in range	36	29	22	2
035jK	Branch to K if (Xj) out of range	36	29	22	2
036jK	Branch to K if (Xj) definite	36	29	22	2
037jK	Branch to K if (Xj) indefinite	36	29	22	2
04ijK	Branch to K if (Bi) = (Bj)	36	29	22	2
05ijK	Branch to K if (Bi) ≠ (Bj)	36	29	22	2
06ijK	Branch to K if (Bi) ≧ (Bj)	36	29	22	2
07ijK	Branch to K if (Bi) < (Bj)	36	29	22	2
10ijx	Transmit (Xj) to Xi	17	10	4	-
11ijk	Logical product of (Xj) and (Xk) to Xi	19	12	6	-
12ijk	Logical sum of (Xj) and (Xk) to Xi	19	12	6	-
13ijk	Logical difference of (Xj) and (Xk) to Xi	19	12	6	-
14ixk	Transmit complement of (Xk) to Xi	17	10	4	-
15ijk	Logical product of (Xj) and complement of (Xk) to Xi	19	12	6	-

TABLE 4-3. CP INSTRUCTION TIMING - MODELS 171 THROUGH 174 (Contd)

Instruction Code	Description	Execution Time (Clock Periods) Model 171	Model 172	Models 173/174	Timing Notes
16ijk	Logical sum of (Xj) and complement of (Xj) to Xi	19	12	6	-
17ijk	Logical difference of (Xj) and complement of (Xk) to Xi	19	12	6	-
20ijk	Left shift (Xi) by jk	19	12	6	-
21ijk	Right shift (Xi) by jk	19	12	6	-
22ijk	Left shift (Xk) nominally (Bj) places to Xi	19	12	6	-
23ijk	Right shift (Xk) nominally (Bj) places to Xi	19	12	6	-
24ijk	Normalize (Xk) to Xi and Bj	20	13	7	-
25ijk	Round normalize (Xk) to Xi and Bj	20	13	7	-
26ijk	Unpack (Xk) to Xi and Bj	19	12	6	-
27ijk	Pack (Xk) and (Bj) to Xi	19	12	6	-
30ijk	Floating sum of (Xj) and (Xk) to Xi	24	17	11	-
31ijk	Floating difference of (Xj) and (Xk) to Xi	24	17	11	-
32ijk	Floating double-precision sum of (Xj) and (Xk) to Xi	24	17	11	-
33ijk	Floating double-precision difference of (Xj) and (Xk) to Xi	24	17	11	-
34ijk	Round floating sum of (Xj) and (Xk) to Xi	24	17	11	-
35ijk	Round floating difference of (Xj) and (Xk) to Xi	24	17	11	-
36ijk	Integer sum of (Xj) and (Xk) to Xi	19	12	6	-
37ijk	Integer difference of (Xj) and (Xk) to Xi	19	12	6	-
40ijk	Floating product of (Xj) and (Xk) to Xi	71	64	58	-
41ijk	Round floating product of (Xj) and (Xk) to Xi	71	64	58	-
42ijk	Floating double-precision product of (Xj) and (Xk) to Xi	71	64	58	-
43ijk	Form mask of jk bits to Xi	19	12	6	-
44ijk	Floating divide (Xj) by (Xk) to Xi	71	64	58	-
45ijk	Round floating divide (Xj) by (Xk) to Xi	71	64	58	-
460xx	Pass	17	10	3	-
464jK	Move indirect	-	-	-	5, 7, 11
465	Move direct	-	-	-	5, 6, 11
466	Compare collated	-	-	-	5, 9, 11
467	Compare uncollated	-	-	-	5, 8, 11
47ixk	Population count of (Xk) to Xi	80	73	67	-
50ijK	Set Ai to (Aj) + K	-	-	-	3
51ijK	Set Ai to (Bj) + K	-	-	-	3
52ijK	Set Ai to (Xj) + K	-	-	-	3
53ijk	Set Ai to (Xj) + (Bk)	-	-	-	3
54ijk	Set Ai to (Aj) + (Bk)	-	-	-	3
55ijk	Set Ai to (Aj) - (Bk)	-	-	-	3
56ijk	Set Ai to (Bj) + (Bk)	-	-	-	3
57ijk	Set Ai to (Bj) - (Bk)	-	-	-	3
60ijK	Set Bi to (Aj) + K	18	11	5	-
61ijK	Set Bi to (Bj) + K	18	11	5	-
62ijK	Set Bi to (Xj) + K	18	11	5	-
63ijk	Set Bi to (Xj) + (Bk)	18	11	5	-
64ijk	Set Bi to (Aj) + (Bk)	18	11	5	-
65ijk	Set Bi to (Aj) - (Bk)	18	11	5	-
66ijk	Set Bi to (Bj) + (Bk)	18	11	5	-
67ijk	Set Bi to (Bj) - (Bk)	18	11	5	-
70ijK	Set Xi to (Aj) + K	19	12	6	-
71ijK	Set Xi to (Bj) + K	19	12	6	-
72ijK	Set Xi to (Xj) + K	19	12	6	-
73ijk	Set Xi to (Xj) + (Bk)	19	12	6	-
74ijk	Set Xi to (Aj) + (Bk)	19	12	6	-
75ijk	Set Xi to (Aj) - (Bk)	19	12	6	-
76ijk	Set Xi to (Bj) + (Bk)	19	12	6	-
77ijk	Set Xi to (Bj) - (Bk)	19	12	6	-

Timing Notes:

1. Instruction placement within a program instruction word may affect the RNI initiation time and the total execution time of the program. (Refer to Instruction Execution - Models 171 through 174 in section 5.)

2. If jump condition not met (model 171, 19 clock periods) (model 172, 12 clock periods) (models 173 and 174, 5 clock periods).
3. If i equals 0 (model 171, 19 clock periods) (model 172, 12 clock periods) (models 173 and 174, 5 clock periods).
   If i equals 1 through 5 (model 171, 35 clock periods) (model 172, 28 clock periods) (models 173 and 174, 21 clock periods).
   If i equals 6 or 7 (model 171, 24 clock periods) (model 172, 17 clock periods) (models 173 and 174, 10 clock periods).
4. Refer to ECS timing information in Volume 3 of publication number 60347100.
5. 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 (PPs, second CP, or ECS).
   c. No conflicts within the instruction or memory refresh conflicts. Memory refresh conflicts may occur only in models A and B.
   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

6. Execution time for model 171:

   T = 2450 + 300N, for N equal to 1 through 4
   T = 1600 + 500N, for N greater than or equal to 5

   Execution time for model 172:

   T = 2100 + 300N, for N equal to 1 through 4
   T = 1250 + 500N, for N greater than or equal to 5

   Execution time for models 173 or 174:

   T = 1750 + 300N, for N equal to 1 through 4
   T = 900 + 500N, for N greater than or equal to 5
7. Execution time for model 172, 173, or 174:

   T = 1000 + move direct instruction execution time (refer to note 5)
8. Execution time for model 171:

   T = 1500 + 725N, if N is even
   T = 1725 + 725N, if N is odd

   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
9. 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
10. When used as error exit, 00 instructions take 52 clock periods
11. In model 171, this instruction is illegal unless compare/move option is installed.

CP INSTRUCTION TIMING — MODEL 175

Execution times for the CP instructions are listed in table 4-4 for model 175. The execution times are listed with the assumption that no conflicts occur. Execution delays result unless all the conditions listed in the timing notes columns exist for the particular instruction. The numbers in the timing notes column refer to notes listed at the end of the table. Execution times are in clock periods, which are each 25 nanoseconds.

TABLE 4-4. CP 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) ≠ 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) ≠ (Bj)	-	26	1, 2, 3, 10, 11, 18
06ijK	Branch to K if (Bi) ≧ (Bj)	-	26	1, 2, 3, 10, 11, 18
07ijK	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 4-4. CP INSTRUCTION TIMING - MODEL 175 (Contd)

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 of (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 difference 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) minus (Xk) to Xi	Floating add	4	8, 12, 13
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

TABLE 4-4. CP INSTRUCTION TIMING - MODEL 175 (Contd)

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
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	Population 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
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

Timing Notes:

1. All previous instruction fetches are complete.
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.

TABLE 4-4. CP INSTRUCTION TIMING - MODEL 175 (Contd)

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 is 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 equals 0, execution time is 2 clock periods, and no storage reference is required. If i equals 1 through 5, execution time is 23 clock periods, and a storage reference is required. If i equals 6 or 7, execution time is 2 clock periods, and a storage reference continues after instruction execution.
17. After the instruction is 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.

CP INSTRUCTION TIMING — MODEL 176

Execution times for CP instructions are listed in table 4-5 for model 176. The execution times are listed with the assumption that no conflicts occur. Execution delays result unless all the conditions listed in the timing notes columns exist for the particular instruction. The numbers in the timing notes column refer to notes listed at the end of the table. Execution times are in clock periods, which are each 27.5 nanoseconds.

TABLE 4-5. CP INSTRUCTION TIMING - MODEL 176

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
00xxx	Error exit to EEA	-	-	-
010xK	Return jump to K	-	13	1, 2, 3
011jk	Block copy (Bj) + K words from LCME to CM	-	(Bj) + K + 22	1, 2, 3, 4, 5, 6, 19, 22
012jK	Block copy (Bj) + K words from CM to LCME	-	(Bj) + K + 13	1, 2, 3, 4, 5, 6, 20, 22
013jK	Exchange exit to (Bj) + K (exit mode flag set)	-	28	1, 2, 3, 4
013xx	Exchange exit to NEA (exit mode flag not set)	-	28	1, 2, 3, 4
014jk	Read LCME at (Xk) to Xj	-	23	5, 7, 8, 9
015jk	Write Xj into LCME at (Xk)	-	3	5, 7, 8, 21
0160k	Reset input channel (Bk) buffer	-	4	8
016jk	Read input channel (Bk) status to Bj (j ≠ 0)	-	3	8
0170k	Reset output channel (Bk) buffer	-	16	8
017jk	Read output channel (Bk) status to Bj (j ≠ 0)	-	3	8
02ixK	Jump to (Bi) + K	-	11	1, 2, 8, 10
030jK	Branch to K if (Xj) = 0	-	11	1, 2, 10, 11
031jK	Branch to K if (Xj) ≠ 0	-	11	1, 2, 10, 11
032jK	Branch to K if (Xj) positive	-	11	1, 2, 10, 11
033jK	Branch to K if (Xj) negative	-	11	1, 2, 10, 11
034jK	Branch to K if (Xj) in range	-	11	1, 2, 10, 11

TABLE 4-5. CP INSTRUCTION TIMING - MODEL 176 (Contd)

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
035jK	Branch to K if (Xj) out of range	-	11	1, 2, 10, 11
036jK	Branch to K if (Xj) definite	-	11	1, 2, 10, 11
037jK	Branch to K if (Xj) indefinite	-	11	1, 2, 10, 11
04ijK	Branch to K if (Bi) = (Bj)	-	11	1, 2, 10, 11
05ijK	Branch to K if (Bi) ≠ (Bj)	-	11	1, 2, 10, 11
06ijK	Branch to K if (Bi) ≧ (Bj)	-	11	1, 2, 10, 11
07ijK	Branch to K if (Bi) < (Bj)	-	11	1, 2, 10, 11
10ijx	Transmit (Xj) to Xi	Boolean	2	2, 8, 12, 13
11ijk	Logical product of (Xj) and (Xk) to Xi	Boolean	2	2, 8, 12, 13
12ijk	Logical sum of (Xj) and (Xk) to Xi	Boolean	2	2, 8, 12, 13
13ijk	Logical difference of (Xj) and (Xk) to Xi	Boolean	2	2, 8, 12, 13
14ixk	Transmit complement of (Xk) to Xi	Boolean	2	2, 8, 12, 13
15ijk	Logical product of (Xj) and complement of (Xk) to Xi	Boolean	2	2, 8, 12, 13
16ijk	Logical sum of (Xj) and complement of (Xk) to Xi	Boolean	2	2, 8, 12, 13
17ijk	Logical difference of (Xj) and complement of (Xk) to Xi	Boolean	2	2, 8, 12, 13
20ijk	Left shift (Xi) by jk	Shift	2	2, 8, 12, 13
21ijk	Right shift (Xi) by jk	Shift	2	2, 8, 12, 13
22ijk	Left shift (Xk) nominally (Bj) places to Xi	Shift	2	2, 8, 12, 13
23ijk	Right shift (Xk) nominally (Bj) places to Xi	Shift	2	2, 8, 12, 13
24ijk	Normalize (Xk) to Xi and Bj	Normalize	3	2, 8, 12, 13
25ijk	Round normalize (Xk) to Xi and Bj	Normalize	3	2, 8, 12, 13
26ijk	Unpack (Xk) to Xi and Bj	Boolean	2	2, 8, 12, 13
27ijk	Pack (Xk) and (Bj) to Xi	Boolean	2	2, 8, 12, 13
30ijk	Floating sum of (Xj) and (Xk) to Xi	Floating add	4	2, 8, 12, 13
31ijk	Floating difference of (Xj) and (Xk) to Xi	Floating add	4	2, 8, 12, 13

TABLE 4-5. CP INSTRUCTION TIMING - MODEL 176 (Contd)

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	2, 8, 12, 13
33ijk	Floating double-precision difference of (Xj) and (Xk) to Xi	Floating add	4	2, 8, 12, 13
34ijk	Round floating sum of (Xj) and (Xk) to Xi	Floating add	4	2, 8, 12, 13
35ijk	Round floating difference of (Xj) and (Xk) to Xi	Floating add	4	2, 8, 12, 13
36ijk	Integer sum of (Xj) and (Xk) to Xi	Long add	2	2, 8, 12, 13
37ijk	Integer difference of (Xj) and (Xk) to Xi	Long add	2	2, 8, 12, 13
40ijk	Floating product of (Xj) and (Xk) to Xi	Multiply	5	2, 8, 12, 13, 14
41ijk	Round floating product of (Xj) and (Xk) to Xi	Multiply	5	2, 8, 12, 13, 14
42ijk	Floating double-precision product of (Xj) and (Xk) to Xi	Multiply	5	2, 8, 12, 13, 14
43ijk	Form mask of jk bits to Xi	Shift	2	2, 8, 12, 13
44ijk	Floating divide (Xj) by (Xk) to Xi	Divide	20	2, 8, 12, 13, 15
45ijk	Round floating divide (Xj) by (Xk) to Xi	Divide	20	2, 8, 12, 13, 15
46xxx	Pass	-	1	-
47ixk	Population count of (Xk) to Xi	Population count	2	2, 8, 12, 13
50ijK	Set Ai to (Aj) + K	Increment	8	2, 8, 16, 17, 18
51ijK	Set Ai to (Bj) + K	Increment	8	2, 8, 16, 17, 18
52ijK	Set Ai to (Xj) + K	Increment	8	2, 8, 16, 17, 18
53ijk	Set Ai to (Xj) + (Bk)	Increment	8	2, 8, 16, 17, 18
54ijk	Set Ai to (Aj) + (Bk)	Increment	8	2, 8, 16, 17, 18
55ijk	Set Ai to (Aj) - (Bk)	Increment	8	2, 8, 16, 17, 18
56ijk	Set Ai to (Bj) + (Bk)	Increment	8	2, 8, 16, 17, 18
57ijk	Set Ai to (Bj) - (Bk)	Increment	8	2, 8, 16, 17, 18
60ijK	Set Bi to (Aj) + K	Increment	2	2, 8, 12, 13
61ijK	Set Bi to (Bj) + K	Increment	2	2, 8, 12, 13
62ijK	Set Bi to (Xj) + K	Increment	2	2, 8, 12, 13
63ijk	Set Bi to (Xj) + (Bk)	Increment	2	2, 8, 12, 13
64ijk	Set Bi to (Aj) + (Bk)	Increment	2	2, 8, 12, 13
65ijk	Set Bi to (Aj) - (Bk)	Increment	2	2, 8, 12, 13
66ijk	Set Bi to (Bj) + (Bk)	Increment	2	2, 8, 12, 13
67ijk	Set Bi to (Bj) - (Bk)	Increment	2	2, 8, 12, 13
70ijK	Set Xi to (Aj) + K	Increment	2	2, 8, 12, 13
71ijK	Set Xi to (Bj) + K	Increment	2	2, 8, 12, 13

TABLE 4-5. CP INSTRUCTION TIMING - MODEL 176 (Contd)

Instruction Code	Description	Functional Unit	Execution Time (Clock Periods)	Timing Notes
72ijK	Set Xi to (Xj) + K	Increment	2	2, 8, 12, 13
73ijk	Set Xi to (Xj) + (Bk)	Increment	2	2, 8, 12, 13
74ijk	Set Xi to (Aj) + (Bk)	Increment	2	2, 8, 12, 13
75ijk	Set Xi to (Aj) - (Bk)	Increment	2	2, 8, 12, 13
76ijk	Set Xi to (Bj) + (Bk)	Increment	2	2, 8, 12, 13
77ijk	Set Xi to (Bj) - (Bk)	Increment	2	2, 8, 12, 13

Timing Notes:

1. All previous instruction fetches are complete.
2. No CM conflicts or SAS backup caused by CM conflicts exist.
3. No I/O word request occurs.
4. All operating registers are free.
5. LCME is not busy.
6. All LCME banks have completed previously initiated read/write cycles.
7. The requested LCME bank has completed a previously initiated read/write cycle.
8. The requested operating register(s) is free.
9. If the requested word is in an LCME bank operand register because of a previous reference, the execution time is 6 clock periods and could be as many as 15 clock periods if bank is busy with a previous instruction, provided no other conflicts occur.
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 is issued to the functional unit, no further delay is possible.
14. The multiply unit is free.
15. The divide unit is free.
16. If no storage reference is required (i is 0), the execution time is 2 clock periods.
17. After the instruction is 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. Execution time 8 refers to read instructions only. With respect to the CPU, a write instruction is completed when Ai is set (2 clock periods). A CM read requires 6 clock periods, and a CM write requires 9 clock periods to complete a bank reference after the increment instruction is in CIW.
19. If the word count is greater than 45 minus W, (W equals the starting word), the execution time is (Bj) plus K plus 26 clock periods.
20. If the transfer does not end with word 17_8, add 17_8 minus W clock periods (W equals the last word transferred).
21. If the requested bank is busy with a previous instruction, execution time could be up to 37 additional clock periods.
22. Models with 524K of LCME have a maximum transfer rate of approximately 32 words per 64 clock periods.

PERIPHERAL PROCESSOR UNIT INSTRUCTIONS — MODEL 176

PERIPHERAL PROCESSOR UNIT INSTRUCTIONS — MODEL 176

Each PPU sequentially executes instructions from its own memory and uses an 18-bit A register for manipulative operations. The A register is the only PPU register used by the programmer. All the PPU arithmetic operations are binary and are performed in a one's complement mode. This mode treats any value of 777777 in the A register as negative zero.

The PPU instructions are the same and produce the same results as the PP instructions, except for the instructions listed in table 4-6. Corresponding PPU and PP instructions produce the same results through the use of different hardware processing techniques. These techniques do not affect the programming and are not evident, except in some of the detailed instruction descriptions that refer to the X register, which exists only in the PPU hardware.

The PPU instruction descriptions provide greater detail than those for the PPs and may be used with corresponding PP descriptions to add clarity. Examples of additional detail are in PPU instructions 01, 02, and 50 through 57. These instruction descriptions are expanded to include special-case explanations which are not repeated in the PP instruction descriptions.

Similarities of the PPU and PP instructions permit common descriptions of the instruction formats, designators, and addressing modes (except for parts of direct and indirect addressing).

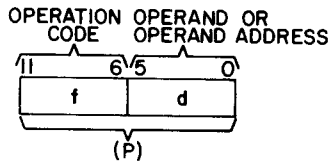
Section 5 contains additional information for PPU and PP programming.

TABLE 4-6. PPU AND PP INSTRUCTION DIFFERENCES

Instruction Code	PPU Instruction Description	PP Instruction Description
00	00xx Error stop	0000 Pass
24	24xx Pass	2400 Pass
25	25xx Pass	2500 Pass
26	26xx Pass	260x Exchange jump 261x Monitor exchange jump - models 171 through 175 262x Monitor exchange jump to MA - models 171 through 175
27	27xx Pass	27X Read program address
60	60dm Jump to m if channel d input word flag set	60d Central read from (A) to d
61	61dm Jump to m if channel d input word flag not set	61dm Central read (d) words from (A) to m
62	62dm Jump to m if channel d input record flag set	62d Central write to (A) from d
63	63dm Jump to m if channel d input record flag not set	63dm Central write (d) words to (A) from m
64	64dm Jump to m if channel d output word flag set	64dm Jump to m if channel d active
65	65dm Jump to m if channel d output word flag not set	65dm Jump to m if channel d inactive
66	66dm Jump to m if channel d output record flag set	66dm Jump to m if channel d full
67	67dm Jump to m if channel d output record flag not set	67dm Jump to m if channel d empty
74	74d Set output record flag on channel d	74d Activate channel d
75	75xx Pass	75d Disconnect channel d
76	76xx Pass	76d Function (A) on channel d
77	77xx Error stop	77dm Function m on channel d

PPU INSTRUCTION FORMATS

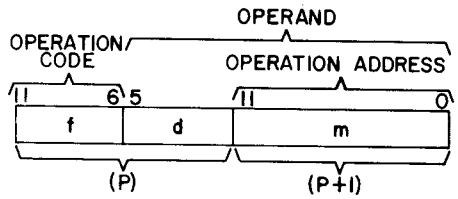
A PPU or PP instruction may have a 12-bit format (figure 4-4) or a 24-bit format (figure 4-5). The 12-bit format has a 6-bit operation code (f) and a 6-bit operand or operand address (d). The 24-bit format uses the 12-bit quantity (m), the content of the next program address (P + 1) with d, or the content of location d to form an 18-bit operand or a 12-bit operand address.



Notes:

1. In direct mode, d is memory address of operand.
2. In indirect mode, d is memory address of operand address.
3. In no address mode, d is a 6-bit operand or shift count.
4. Shaded areas, not shown, are unused.

Figure 4-4. PPU/PP 12-Bit Instruction Format



Notes:

1. In indexed mode, d is index address for modifying operand address, m is base address of operand, and (d) + m is operand address.
2. In constant mode, dm is an 18-bit operand.
3. Shaded areas, not shown, are unused.

Figure 4-5. PPU/PP 24-Bit Instruction Format

PPU INSTRUCTION DESIGNATORS

Table 4-7 lists the PPU and PP instruction designators and their uses.

TABLE 4-7. PPU AND PP INSTRUCTION DESIGNATORS

Designator	Use
f	6-bit operation code
d	6-bit operand or address
m	12-bit operand or address
fd	12-bit instruction code
dm	18-bit operand
x	Unused register
A	Arithmetic register
P	Program register
Q	Q register
( )	Content of a register or location
(( ))	Indirect addressing which specifies the content of a location whose address is specified by a designator inside the parentheses

PPU INSTRUCTION ADDRESSING MODES

Several addressing modes permit PPU and PP program indexing and the manipulation of operands. The addressing modes consist of no address, constant address, direct address, indirect address, and indexed direct address. These modes are summarized in table 4-8.

No Address

In this mode, the PPUs and PPs directly use d as an operand. This mode eliminates the need for storing constants. The d quantity is considered as an 18-bit number, the upper 12-bits of which are zero.

Constant Address

In this mode, the PPUs and PPs directly use dm as an operand. This mode eliminates the need for storing constants. The dm quantity uses d as the upper 6 bits and m as the lower 12 bits of an 18-bit constant.

Direct Address

In this mode, the PPUs and PPs use d as the address of the operand. The d quantity specifies one of the first 64 addresses (0000 through 0077, octal) in PPM.

Indirect Address

In this mode, the PPUs and PPs use d to specify an address in which the content is the address of the desired operand. Thus, d specifies the operand address indirectly. An indirect or an indexed direct address requires one more PPM reference than a direct address.

In the PPs, an address of 7777 (octal) is accessible if the content of the operand is 7777 (octal).

Indexed Direct Address

In this mode, the PPUs and PPs use the content of d plus m as the address of the operand. The d quantity specifies the content of one of the first 63 memory addresses (0001 through 0077, octal). The m quantity is a base address that adds to the content of d to form a 12-bit address. If d is nonzero, the content of address d plus m produce the 12-bit address. If d is zero, m is the operand address.

In the PPUs, the 12-bit address may reference any of the possible memory addresses (0000 through 7776, octal). The PPUs cannot reference address 7777 (octal).

In the PPs, the 12-bit address is specified by d and m (expressed octally) as follows:

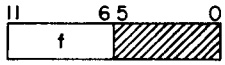
	m=0	m=7777	m=0<m<7777
d=0	0	0	m
d≠0, (d)=0	0	0	m
d≠0, (d)=7777	0	7777	m
d≠0, 0<(d)<7777	(d)	(d)	M+(d)

In the PP block I/O and CM access instructions, d has an alternate meaning and is not used in address computation. The first word address for these instructions is formed directly from m and can reference location 7777. The order of reference is 7777-0000-0001-0002-0003.

TABLE 4-8. PPU AND PP INSTRUCTION ADDRESSING MODES

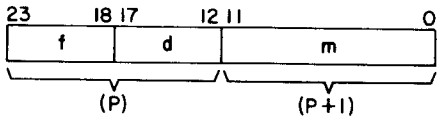
Instruction Type	Addressing Mode				
	No Address	Constant	Direct	Indirect	Indexed Direct
Load	14	20	30	40	50
Add	16	21	31	41	51
Subtract	17	-	32	42	52
Logical difference	11	23	33	43	53
Store	-	-	34	44	54
Replace add	-	-	35	45	55
Replace add one	-	-	36	46	56
Replace subtract one	-	-	37	47	57
Long jump	-	-	-	-	01
Return jump	-	-	-	-	02
Unconditional jump	03	-	-	-	-
Zero jump	04	-	-	-	-
Nonzero jump	05	-	-	-	-
Positive jump	06	-	-	-	-
Negative jump	07	-	-	-	-
Shift	10	-	-	-	-
Logical product	12	22	-	-	-
Selective clear	13	-	-	-	-
Load complement	15	-	-	-	-

00xx Error Stop



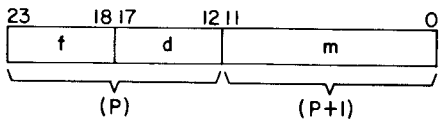
This instruction causes the PPU program execution to stop and to indicate a program error condition to the status and control register. The PPU can be restarted only by a deadstart.

0100m Long Jump to m



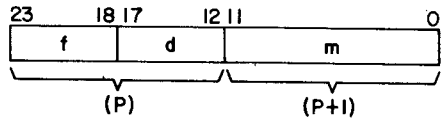
This instruction terminates the current program sequence with a jump to a new sequence beginning at address m. The value of d must be zero for this instruction. The instruction begins by reading the quantity m from the storage location determined by the content of P plus 1 to the X register. The address for the new program sequence forms by adding the content of X to a zero value in the Q register. This address is then used to obtain the first word of the new program sequence.

01dm Long Jump to m + (d)



This instruction terminates the current program sequence with a jump to a new sequence beginning at address m plus the content of d. The value of d must be nonzero for this instruction. The instruction begins by reading the quantity m from the storage location determined by the content of P plus 1 and holding this quantity in the Q register. The content of location d then reads into the X register. The address for the new program sequence forms by adding the content of Q to the content of X in a 12-bit one's complement mode. This address is then used to obtain the first word of the new program sequence.

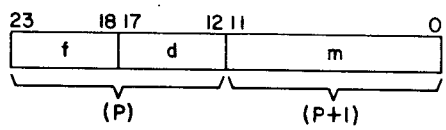
0200m Return Jump to m



This instruction interrupts the current program sequence and inserts the execution of a subroutine between the current instruction in the present sequence and the following instruction. The called subroutine must have a common exit point in the form of a long jump to m instruction preceding the entry point. The return jump instruction inserts the exit address in the m location of the subroutine exit and then jumps to the entry point in the following word.

The value of d in this instruction must be zero. The instruction begins by reading the quantity m from the storage location determined by the content of P plus 1 to the Q register. This quantity is then used as an address to store the content of P plus 2 at storage location m. The first word of the new program sequence then reads from storage location m plus 1.

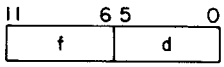
02dm Return Jump to m + (d)



This instruction interrupts the current program sequence and inserts the execution of a subroutine between the current instruction in the present sequence and the following instruction. The called subroutine must have a common exit point in the form of a long jump to m instruction preceding the entry point. The return jump instruction inserts the exit address in the m location of the subroutine exit and then jumps to the entry point in the following word.

The value of d in this instruction must be nonzero. The instruction begins by reading the quantity m from the storage location determined by the content of P plus 1 to the Q register. The content of location d then reads into the X register. The address for the new program sequence forms by adding the content of Q to the content of X in a 12-bit one's complement mode. The resulting address is used to store the content of P plus 2 in the m field of the called subroutine exit instruction. The first word of the new program then reads from the following storage location.

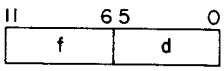
03d Unconditional Jump d



This instruction interrupts the current program sequence with a jump to a new sequence beginning at an address incrementally related to the current program address. The d designator may specify a new sequence which begins at an address either forward or backward from the current address by an amount no greater than 31 (decimal) locations. The d designator is considered as a 6-bit one's complement number in determining the increment for the jump.

As an example, consider a d value of 16 (octal). The new program sequence in this case begins with an instruction word located 16 (octal) locations beyond the location of the 03d instruction. Now consider a d value of 55 (octal). The new program sequence in this case begins with an instruction word located 22 (octal) locations before the location of the 03d instruction. Values of 00 and 77 for the d designator must not be used with this instruction. These two values cause the PPU program to lock up and require deadstarting the system with a new program.

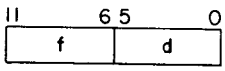
04d Zero Jump d



This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the content of the A register. If the content of A is 000000, the current program sequence terminates with a jump to an address specified by the content of P plus the d designator. If the content of A is not 000000, the current program sequence continues with the execution of the next instruction. When the value of the content of A is 777777, it is not considered as zero for this instruction.

If the jump occurs, the new program sequence begins at an address either forward or backward from the current address by an amount not greater than 31 (decimal) locations. The d designator is considered as a 6-bit one's complement number in determining the increment for the jump. (Refer to instruction 03d for examples.)

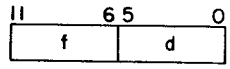
05d Nonzero Jump d



This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the content of the A register. If the content of A is not 000000, the current program sequence terminates with a jump to an address specified by the content of P plus the d designator. If the content of A is 000000, the current program sequence continues with the execution of the next instruction. When the value of the content of A is 777777, it is not considered as zero for this instruction.

If the jump occurs, the new program sequence begins at an address either forward or backward from the current address by an amount not greater than 31 (decimal) locations. The d designator is considered as a 6-bit one's complement number in determining the increment for the jump. (Refer to instruction 03d for examples.)

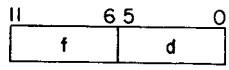
06d Plus Jump d



This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the content of the A register. If the highest-order bit in A has a zero value, the current program sequence terminates with a jump to an address specified by the content of P plus the d designator. If the highest-order bit in A has one value, the current program sequence continues with the execution of the next instruction.

If the jump occurs, the new program sequence begins at an address either forward or backward from the current address by an amount not greater than 31 (decimal) locations. The d designator is considered as a 6-bit one's complement number in determining the increment for the jump. (Refer to instruction 03d for examples.)

07d Minus Jump d

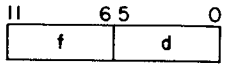


This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the content of the A register. If the highest-order bit in A has a one value, the current program sequence terminates with a jump to an address specified by the content of P plus the d designator. If the highest-order bit in A has a zero value, the current program sequence continues with the execution of the next instruction.

If the jump occurs, the new program sequence begins at an address either forward or backward from

the current address by an amount no greater than 31 (decimal) locations. The d designator is considered as a 6-bit one's complement number in determining the increment for the jump. (Refer to instruction 03d for examples.)

10d Shift (A) by d

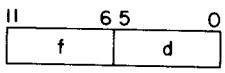


This instruction shifts the content of the A register either to the right open-ended or to the left circularly as specified by the d designator. The d designator is treated as a 6-bit one's complement number. If the highest-order bit in the d designator is zero, the content of A shifts circularly to the left by the number of bit positions indicated in the value of the d designator. If the highest-order bit in the d designator is one, the content of A shifts open-ended to the right by the complement of the value of the d designator.

In a left circular shift, the content of A shifts one bit position at a time. In each shift, the lowest-order bit position in the register fills by the bit previously held in the highest-order bit position. Bits are not lost in this process but are repositioned toward the higher-order positions. A d designator value of 00 causes no shift. A d designator value greater than 18 (decimal) causes the content of A to shift completely around the register. A maximum of 31 (decimal) shifts counts may be used.

In a right open-ended shift, the content of A shifts one bit position at a time toward the lower-order bit positions in the register. The highest-order bit position in A fills with a zero value as each shift occurs. The lowest order bit in A discards as each shift occurs. A maximum of 31 (decimal) shift counts may be used. For all shift counts larger than 17 (decimal), the final A register value is 000000. A designator value of 77 causes no shift to take place.

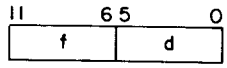
11d Logical Difference (A) and d



This instruction forms the logical difference of the original content of A and the d designator, considered as a 6-bit positive integer, in the A register. The highest-order 12 bits in A are not affected by the operation.

The logical difference is the result of a bit-by-bit comparison of the two binary quantities. If two corresponding bits are equal, the resulting bit is zero; if unequal, the result is one.

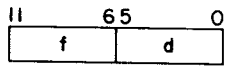
12d Logical Product (A) and d



This instruction forms the logical product of the original content of A and the d designator, considered as a 6-bit positive integer, in the A register. The highest-order 12 bits in A are always cleared to zero by this instruction.

The logical product is the result of a bit-by-bit comparison of the two binary quantities. If two corresponding bits are ones, the resulting bit is one; if not, the result is zero.

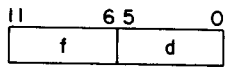
13d Selective Clear (A) by d



This instruction forms the logical product of the original content of A and the complement of the d designator, considered as a 6-bit positive integer, in the A register. The highest-order 12 bits in A are not affected by this instruction.

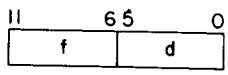
The selective clear is a bit-by-bit comparison of the two binary quantities. Any of the lower six bits in A clear if the corresponding bits of d set.

14d Load d



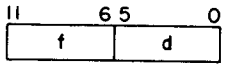
This instruction enters a copy of the d designator, considered as a 6-bit positive integer, into the A register. The highest-order 12 bits in A always clear to zero by this instruction.

15d Load Complement d



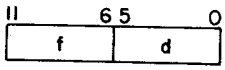
This instruction enters a complemented copy of the d designator into the A register. The highest-order 12 bits in A always set to one by this instruction. The lowest-order six bits are bit-by-bit complements of the corresponding bits in the d designator.

16d Add (A) + d



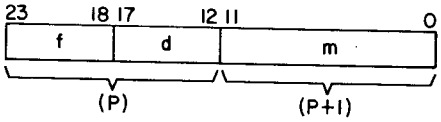
This instruction adds the d designator, considered as a 6-bit positive quantity, to the current content of the A register. The result remains in A. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from the d designator by adding 12 higher-order zero bits.

17d Subtract (A) — d



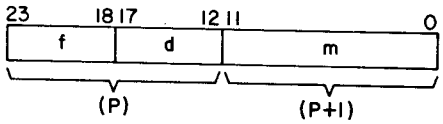
This instruction subtracts the d designator, considered as a 6-bit positive quantity, from the current content of the A register. The result remains in A. An 18-bit operand forms from the d designator. This operand consists of 12 one bits in the highest-order bit positions and 6 lowest-order bits which are bit-by-bit complements of the corresponding bits in the d designator. This 18-bit operand adds to the original content of A in an 18-bit one's complement mode.

20dm Load dm



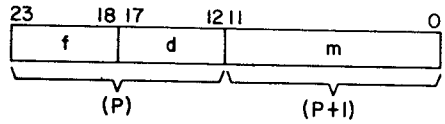
This instruction clears the A register and enters an 18-bit operand, consisting of the d and m designators. The d designator inserts into the highest-order 6-bit positions, and the m designator inserts into the lowest-order 12-bit positions.

21dm Add (A) + dm



This instruction adds an 18-bit operand consisting of the d and m designators to the current content of the A register. The result remains in A. The addition is in an 18-bit one's complement mode. The d designator forms the highest-order 6 bits, and the m designator completes the lowest-order 12 bits.

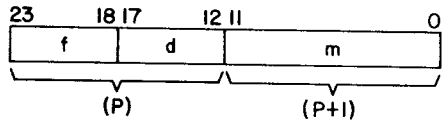
22dm Logical Product (A) and dm



This instruction forms the logical product of the content of the A register and an 18-bit operand consisting of the d and m designators. The result remains in A. The d designator forms the highest-order 6 bits, and the m designator completes the lowest-order 12 bits.

The logical product is the result of a bit-by-bit comparison of the two binary quantities. If two corresponding bits are ones, the resulting bit is one; if not, the result is zero.

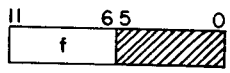
23dm Logical Difference (A) and dm



This instruction forms the logical difference of the content of the A register and an 18-bit operand consisting of the d and m designators. The result remains in A. The d designator forms the highest-order 6 bits, and the m designator completes the lowest-order 12 bits.

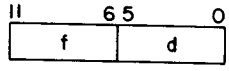
The logical difference is the result of a bit-by-bit comparison of the two binary quantities. If two corresponding bits are equal, the resulting bit is zero; if unequal, the result is one.

24xx through 27xx Pass



These four instructions are identical and perform no logical function. Each instruction results in a 5-clock-period delay.

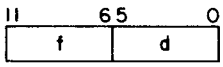
30d Load (d)



This instruction clears the A register and enters a 12-bit operand from location d. The operand enters

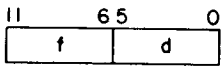
into A as a 12-bit positive integer. The highest-order six bits in A always clear by this instruction.

31d Add (A) + d



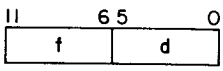
This instruction adds the content of location d, considered as a 12-bit positive quantity, to the current content of the A register. The result remains in A. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from location d by adding six higher-order zero bits.

32d Subtract (A) — (d)



This instruction subtracts the content of location d, considered as a 12-bit positive quantity, from the current content of the A register. The result remains in A. The operation adds the complement of location d to the content of A in an 18-bit one's complement mode. An 18-bit operand forms for the addition by forcing the highest-order six bits to a one value. The lowest-order 12 bits are the bit-by-bit complement of location d.

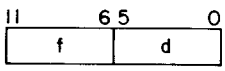
33d Logical Difference (A) and (d)



This instruction forms, in the A register, the logical difference of the content of location d, considered as a 12-bit positive quantity, and the original content of A. The highest-order six bits in A are not affected by this operation.

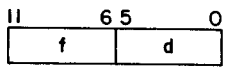
The logical difference is the result of a bit-by-bit comparison of the two binary quantities. If any corresponding bits are equal, the resulting bit is zero; if unequal, the result is one.

34d Store (A) at (d)



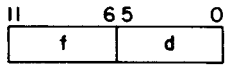
This instruction stores the lowest-order 12 bits of the content of the A register in location d. The content of A is not altered in this process.

35d Replace Add (A) + (d)



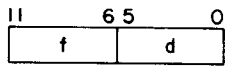
This instruction adds the content of location d, considered as a 12-bit positive quantity, to the current content of the A register. The result remains in A and also stores in location d. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from location d by adding six higher-order zero bits. The result stored in location d is the lowest-order 12 bits of the resulting 18-bit sum.

36d Replace Add One (d)



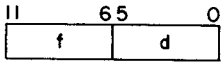
This instruction increases the content of location d by one count. Execution begins by clearing the A register and entering a value of plus one. The content of location d reads from storage to the X register and then adds to the content of A in an 18-bit one's complement mode. The location d value is treated as a 12-bit positive quantity in this process. An 18-bit operand forms from the content of X by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits store in location d. The arithmetic is essentially two's complement as viewed by location d, and the quantity in A is not necessarily equal to the result in location d.

37d Replace Subtract One (d)



This instruction decreases the content of location d by one count. Execution begins by clearing the A register and entering a value of minus one. The content of location d reads from storage to the X register and then adds to the content of A in an 18-bit one's complement mode. The location d value is treated as a 12-bit positive quantity in this process. An 18-bit operand forms from the content of X by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits store in location d. The arithmetic is essentially two's complement as viewed by location d, and the quantity in A is not necessarily equal to the result in location d.

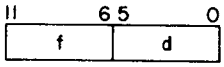
40d Load ((d))



This instruction clears the A register and enters a 12-bit operand from storage. The highest-order six bits in A always clear by this instruction.

Instruction executes begins with a storage reference to location d. The content of this location reads into the X register. A second storage reference is then made using the content of X as the storage address. This operand reads into A, and the highest-order six bits in A clear. A third storage reference then reads the next instruction word.

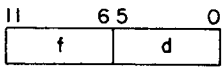
41d Add (A) + ((d))



This instruction reads an operand from storage and adds it to the current content of the A register. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The address for the operand is in location d.

Instruction execution begins with a storage reference to location d. The content of this location reads into the X register. A second storage reference is then made using the content of X as the storage address. This operand reads into X and then adds to the content of A. A third storage reference then reads the next instruction word.

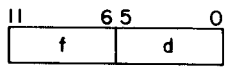
42d Subtract (A) — ((d))



This instruction reads an operand from storage and subtracts it from the current content of the A register. The result remains in A. The address for the operand is in location d. The operation performs by adding the complement of the operand to the content of A in an 18-bit one's complement mode. An 18-bit operand for the addition forms from the 12-bit storage operand by forcing the highest-order six bits to a one value. The lowest-order 12 bits are the bit-by-bit complement of the storage operand values.

Instruction execution begins with a storage reference to location d. The content of this location reads into the X register. A second storage reference then occurs using the content of X as the storage address. This operand reads into X and then subtracts from the content of A. A third storage reference then reads the next instruction word.

43d Logical Difference (A) and ((d))

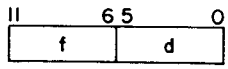


This instruction forms the logical difference of an operand read from storage and the original content of A in the A register. The highest-order six bits in A are not affected by this operation. The storage address for the operand is in location d.

The logical difference is the result of a bit-by-bit comparison of the two binary quantities. If the corresponding bits are equal, the resulting bit is zero; if unequal, the result is one.

Instruction execution begins with a storage reference to location d. The content of this location reads into the X register. A second reference to storage occurs using the content of X as the storage address. This operand reads into X, and the logical difference then forms and enters into A. A third storage reference then reads the next instruction word.

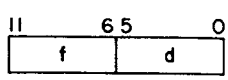
44d Store (A) at ((d))



This instruction stores the lowest-order 12 bits of the content of the A register in a storage location specified by the content of location d. The content of A is not altered in this process.

Execution begins with a storage reference to location d. The content of this location reads into the X register. A second reference to storage occurs using the content of X as the storage address. The data read from storage discards in this reference, and the lowest-order 12 bits of the content of A store. A third storage reference then reads the next instruction word.

45d Replace Add (A) + ((d))

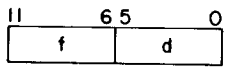


This instruction reads an operand from storage and adds it to the current content of the A register. The

result remains in A and also stores in the same memory location from which the operand was read. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result returned to storage is the lowest-order 12 bits of the final content of A. The storage address for reading the operand and storing the result is in location d. The result stored is not necessarily equal to the result remaining in A.

Four storage references are required in the execution of this instruction. The first reference reads the content of location d into X and then into Q. A second storage reference is to the content of X for the storage address. This operand reads into X and then adds to the content of A. A third storage reference stores the lowest-order 12 bits of the resulting sum using the content of Q as the storage address. The fourth storage reference then reads the next instruction word.

46d Replace Add One ((d))

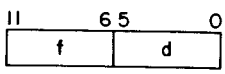


This instruction reads an operand from storage, increases its value by one count, and returns the result to the same storage location. The storage address for reading the operand and storing the result is in location d. The result remains in the A register and in storage.

Execution begins by clearing A and entering a value of plus one. The operand then reads from storage and adds to the content of A in an 18-bit one's complement mode. The operand is treated as a 12-bit positive quantity in this process. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits return to storage. The arithmetic is essentially two's complement as viewed from storage, and the quantity in A is not necessarily equal to the result in storage.

Four storage references are required in the execution of this instruction. The first reference reads the content of location d into X and then into Q. A second storage reference is made using the content of X as the storage address. This operand reads into X and then adds to the content of A. A third storage reference stores the lowest-order 12 bits of the resulting sum using the content of Q as the storage address. The fourth storage reference then reads the next instruction word.

47d Replace Subtract One ((d))

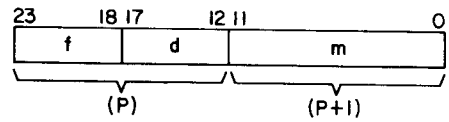


This instruction reads an operand from storage, decreases its value by one count, and returns the result to the same storage location. The storage address for reading the operand and storing the result is in location d. The result remains in A as well as in storage.

Execution begins by clearing A and entering a value of minus one. The operand then reads from storage and adds to the content of A in an 18-bit one's complement mode. The operand is treated as a 12-bit positive quantity in this process. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits return to storage. The arithmetic is essentially two's complement as viewed from storage, and the quantity in A is not necessarily equal to the result in storage.

Four storage references are required in the execution of this instruction. The first reference reads the content of location d into the Q register. A second storage reference is made using the content of the X register as the storage address. This operand reads into X and then adds to the content of A. A third storage reference stores the lowest-order 12 bits of the resulting sum using the content of Q as the storage address. The fourth storage reference then reads the next instruction word.

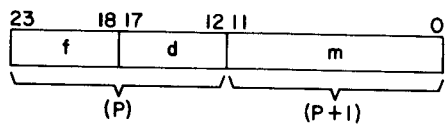
5000m Load (m)



This instruction clears the A register and enters a 12-bit operand from storage. The address for the operand is in the m designator for this instruction. The operand enters A as a 12-bit positive integer. The highest-order six bits in A always clear.

Instruction execution begins with a storage reference for the m designator. This quantity reads into the X register. A second storage reference then occurs, using the content of X as the storage address. This operand reads into X and then enters A. A third storage reference then reads the next instruction word.

50dm Load (m + (d))

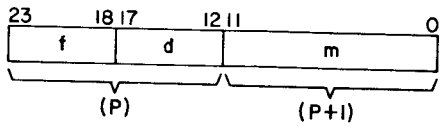


This instruction clears the A register and enters a 12-bit operand from storage. The address for the

operand forms by adding the m designator and the content of location d in a 12-bit one's complement mode. The operand enters A as a 12-bit positive integer. The highest-order six bits in A always clear. The d designator must have a nonzero value for this instruction.

Four storage references are required in the execution of this instruction. The first reference reads the m designator into the X register and then into the Q register. The second reference reads the content of location d into X. The third reference uses the content of Q plus the content of X as a storage address for the operand. This quantity reads into X and then enters A. The fourth storage reference then reads the next instruction word.

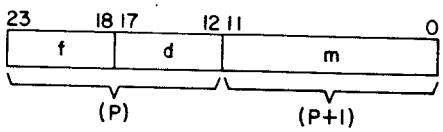
5100m Add (A) + (m)



This instruction reads an operand from storage and adds it to the current content of the A register. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The storage address for the operand is in the m designator for this instruction.

Instruction execution begins with a storage reference for the m designator. This quantity reads into the X register. A second storage reference then occurs using the content of X as the storage address. This operand reads into X and then enters A. A third storage reference then reads the next instruction word.

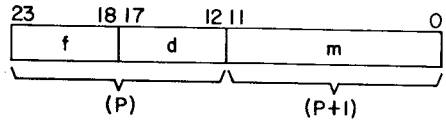
51dm Add (A) + (m + (d))



This instruction reads an operand from storage and adds it to the current content of the A register. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The storage address for the operand forms by adding the m designator and the content of location d in a 12-bit one's complement mode. The d designator must have a nonzero value for this instruction.

Four storage references are required in the execution of this instruction. The first reference reads the m designator into the X register and then into the Q register. The second reference reads the content of location d into X. The third reference uses the content of Q plus the content of X as a storage address for the operand. This quantity reads into X and then enters A. The fourth storage reference then reads the next instruction word.

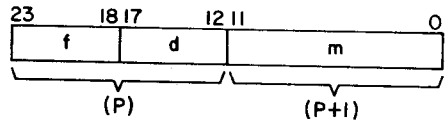
5200m Subtract (A) — (m)



This instruction reads an operand from storage and subtracts it from the current content of the A register. The result remains in A. The storage address for the operand is in the m designator for this instruction. The operation adds the complement of the operand to the content of A in an 18-bit one's complement mode. An 18-bit operand for the addition forms from the 12-bit storage operand by forcing the highest-order six bits to a one value. The lowest-order 12 bits are the bit-by-bit complement of the storage operand values.

Instruction execution begins with a storage reference for the m designator. This quantity reads into the X register. A second storage reference then uses the content of X as the storage address. This operand is read into X and then subtracted in A. A third storage reference then reads the next instruction word.

52dm Subtract (A) — (m + (d))

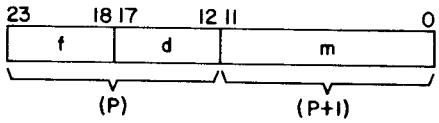


This instruction reads an operand from storage and subtracts it from the current content of the A register. The result remains in A. The address for the operand forms by adding the m designator and the content of location d in a 12-bit one's complement mode. The arithmetic operation adds the complement of the operand to the content of A in an 18-bit one's complement mode. An 18-bit operand for the addition forms the 12-bit storage operand by forcing the highest-order six bits to a value of one. The lowest-order 12 bits are the bit-by-bit complement of the storage operand values. The d designator must have a nonzero value for this instruction.

Four storage references are required in the execution of this instruction. The first reference reads the m designator into the X register and then into the Q register. The second reference reads the content of location d into X. The third reference uses the content of Q plus the content of X as a

storage address for the operand. This quantity is read into X and then subtracted in A. The fourth storage reference then reads the next instruction word.

5300m Logical Difference (A) and (m)

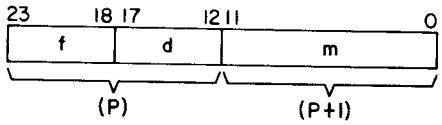


This instruction forms the logical difference of an operand read from storage and the original content of A in the A register. The highest-order six bits in A are not affected by this operation. The storage address for the operand is in the m designator for this instruction.

The logical difference is the result of a bit-by-bit comparison of the two binary quantities. If two corresponding bits are equal, the resulting bit is zero; if unequal, the result is one.

Instruction execution begins with a storage reference for the m designator. This quantity reads into the X register. A second storage reference then uses the content of X as the storage address. This operand reads into X and the logical difference enters A. A third storage reference then reads the next instruction word.

53dm Logical Difference (A) and (m + (d))



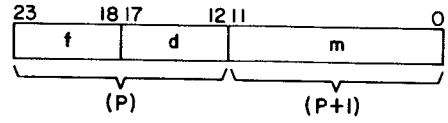
This instruction forms the logical difference of an operand read from storage and the original content of A in the A register. The highest-order six bits in A are not affected by this operation. The address for the operand forms by adding the m designator and the content of location d in a 12-bit one's complement mode. The d designator must have a nonzero value for this instruction.

The logical difference is the result of a bit-by-bit comparison of the two binary quantities. If two corresponding bits are equal, the resulting bit is zero; if unequal, the result is one.

Four storage references are required in the execution of this instruction. The first reference reads the m designator into the X register and then into the Q register. The second reference reads the content of location d into X. The third reference uses the content of Q plus the content of X as a storage address for the operand. This quantity reads into X and the logical difference enters in A.

The fourth storage reference then reads the next instruction word.

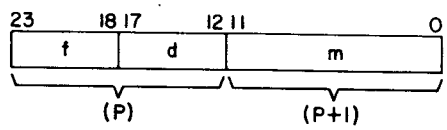
5400m Store (A) at (m)



This instruction stores the lowest-order 12 bits of the content of the A register in a storage location specified by the m designator. The content of A is not altered in this process.

Execution begins with a storage reference for the m designator. This quantity reads into the X register. A second storage reference uses the content of X as the storage address. The lowest-order 12 bits of the content of A store during the storage cycle. A third storage reference then reads the next instruction word.

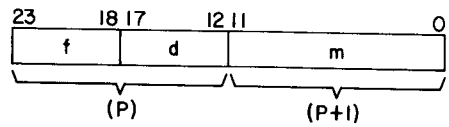
54dm Store (A) at (m + (d))



This instruction stores the lowest-order 12 bits of the content of the A register. The storage address forms by adding the m designator and the content of location d in a 12-bit one's complement mode. The d designator must have a nonzero value for this instruction.

Four storage references are required in the execution of this instruction. The first reference reads the m designator into the X register and then into the Q register. The second reference reads the content of location d into X. The third reference uses the content of Q plus the content of X as a storage address for storing the lowest-order 12 bits of A. The fourth storage reference reads the next instruction word.

5500m Replace Add (A) +(m)

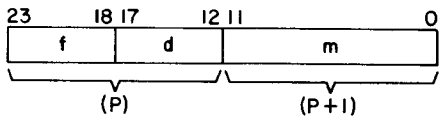


This instruction reads an operand from storage and adds it to the current content of the A register.

The result remains in A and also stores in the same memory location from which the operand was read. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result returned to storage is the lowest-order 12 bits of the final content of A. The storage address for reading the operand and storing the result in the m designator for this instruction. The result stored is not necessarily equal to the result remaining in A.

Four storage references are required in the execution of this instruction. The first reference reads the m designator into the X register and the Q register. A second reference uses the content of X as the storage address. This operand reads into X and adds into A. A third reference stores the lowest-order 12 bits of the content of A using the content of Q as the storage address. The fourth reference then reads the next instruction word.

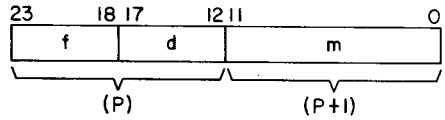
55dm Replace Add (A) + (m + (d))



This instruction reads an operand from storage and adds it to the current content of the A register. The result remains in A and also stores in the same memory location from which the operand was read. The addition is in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result returned to storage is the lowest-order 12 bits of the final content of A. The storage address for reading the operand and storing the result forms by adding the m designator to the content of location d in a 12-bit one's complement mode. The result stored is not necessarily equal to the result remaining in A.

Five storage references are required in the execution of this instruction. The first reference reads the m designator into the X register and then into the Q register. The second reference reads the content of location d into X. The third reference uses the content of Q plus the content of X to read the operand into X. The addition occurs in A. The content of Q plus the content of X enters Q at this same time. The fourth storage reference stores the lowest order 12 bits of the content of A using the new content of Q as a storage address. The last storage reference then reads the next program instruction word.

5600m Replace Add One (m)

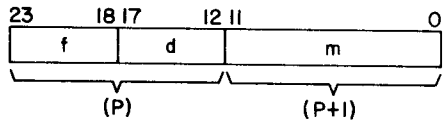


This instruction reads an operand from storage, increases its value by one count, and returns the result to the same storage location. The storage address for reading the operand and storing the result is in the m designator for this instruction. The result remains in A as well as in storage.

Execution begins by clearing A and entering a value of plus one. The operand then reads from storage and adds to the content of A in an 18-bit one's complement mode. The operand is treated as a 12-bit positive quantity in this process. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits return to storage. The arithmetic is essentially two's complement as viewed from storage, and the quantity in A is not necessarily equal to the result in storage.

Four storage references are required in the execution of this instruction. The first reference reads the m designator from storage into the X register and then into the Q register. A second storage reference uses the content of X as the storage address. This operand reads into X and adds into A. A third reference stores the lowest-order 12 bits of the content of A using the content of Q as the storage address. The fourth reference then reads the next instruction word.

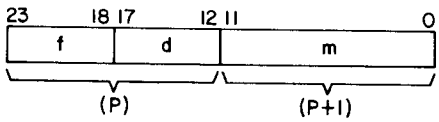
56dm Replace Add One (m + (d))



This instruction reads an operand from storage, increases its value by one count, and returns the result to the same storage location. The storage address for reading the operand and storing the result forms by adding the m designator to the content of location d in a 12-bit one's complement mode. The result remains in the A register and in storage.

Execution begins by clearing A and entering a value of plus one. The m designator reads from storage and enters the X register and then the Q register. A second storage reference reads the content of location d into X. A third reference reads the operand into X using the content of Q plus the content of X as the storage address. The content of Q plus the content of X enters Q at this same time. The operand then adds into A in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits are returned to storage using the content of Q as the storage address. The arithmetic is essentially two's complement as viewed from storage, and the quantity in A is not necessarily equal to the result in storage. A fifth storage reference then reads the next instruction word.

5700m Replace Subtract One (m)

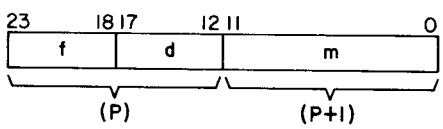


This instruction reads an operand from storage, decreases its value by one count, and returns the result to the same storage location. The storage address for reading the operand and storing the result is in the m designator for this instruction. The result remains in the A register and in storage.

Execution begins by clearing A and entering a value of minus one. The operand then reads from storage and adds to the content of A in an 18-bit one's complement mode. The operand is treated as a 12-bit positive quantity in this process. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits return to storage. The arithmetic is essentially two's complement as viewed from storage, and the quantity in A is not necessarily equal to the result in storage.

Four storage references are required in the execution of this instruction. The first reference reads the m designator from storage into the X register and then into the Q register. A second storage reference is made using the content of X as the storage address. This operand reads into X and adds into A. A third reference stores the lowest-order 12 bits of the content of A using the content of Q as the storage address. The fourth reference then reads the next instruction word.

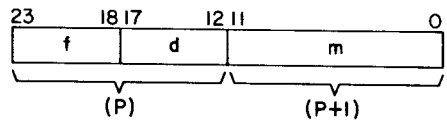
57dm Replace Subtract One (m + (d))



This instruction reads an operand from storage, decreases its value by one count, and returns the result to the same storage location. The storage address for reading the operand and storing the result forms by adding the m designator to the content of location d in a 12-bit one's complement mode. The result remains in the A register and in storage.

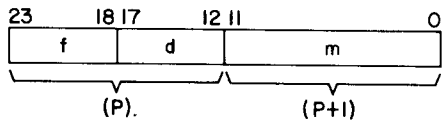
Execution begins by clearing A and entering a value of minus one. The m designator reads from storage and enters the X register and then the Q register. A second storage reference reads the contents of location d into X. A third reference reads the operand into X using the content of Q plus the content of X as the storage address. The content of Q plus the content of X enters Q at the same time. The operand then adds into A in an 18-bit one's complement mode. An 18-bit operand forms from the 12-bit storage operand by adding six higher-order zero bits. The result remains in A, and the lowest-order 12 bits return to storage using the content of Q as the storage address. The arithmetic is essentially two's complement as viewed from storage, and the quantity in A is not necessarily equal to the result in storage. A fifth storage reference then reads the next instruction word.

60dm Jump to m if Channel d Input Word Flag Set



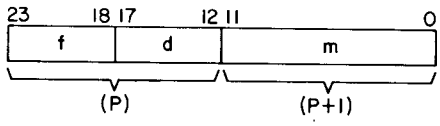
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d input word flag. A new program sequence initiates beginning at address m if the channel d input word flag is set. The current program sequence continues if the flag is not set.

61dm Jump to m if Channel d Input Word Flag Not Set



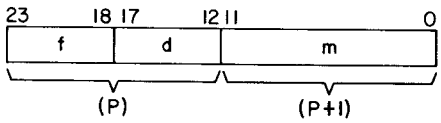
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d input word flag. A new program sequence initiates beginning at address m if the channel d input word flag is not set. The current program sequence continues if the flag is set.

62dm Jump to m if Channel d Input Record Flag Set



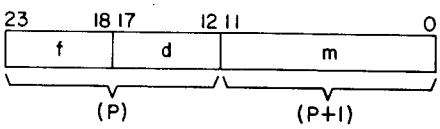
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d input record flag. A new program sequence initiates beginning at address m if the channel d input record flag is set. The current program sequence continues if the flag is not set.

63dm Jump to m if Channel d Input Record Flag Not Set



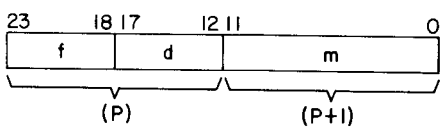
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d input record flag. A new program sequence initiates beginning at address m if the channel d input record flag is not set. The current program sequence continues if the flag is set.

64dm Jump to m if Channel d Output Word Flag Set



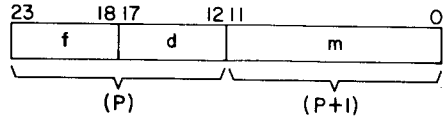
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d output word flag. A new program sequence initiates beginning at address m if the channel d output word flag is set. The current program sequence continues if the flag is not set.

65dm Jump to m if Channel d Output Word Flag Not Set



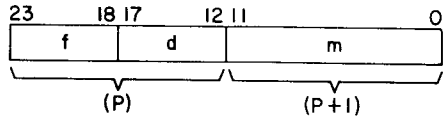
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d output word flag. A new program sequence initiates beginning at address m if the channel d output word flag is not set. The current program sequence continues if the flag is set.

66dm Jump to m if Channel d Output Record Flag Set



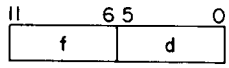
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d output record flag. A new program sequence initiates beginning at address m if the channel d output record flag is set. The current program sequence continues if the flag is not set.

67dm Jump to m if Channel d Output Record Flag Not Set



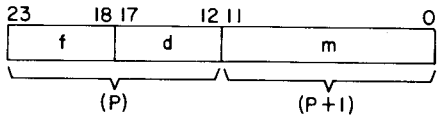
This conditional branch instruction continues the current program sequence or jumps to a new program sequence, depending upon the condition of the channel d output record flag. A new program sequence initiates beginning at address m if the channel d output record flag is not set. The current program sequence continues if the flag is set.

70d Input to A from Channel d



This instruction reads one word from input channel d and enters the word in the A register. This instruction does not execute until the channel d input word flag is set. If the flag is not set at the time the instruction reads from storage, the PPU program stops with the instruction in the fd register and waits until the flag is set by an external signal. The channel d input record flag does not affect execution of this instruction. This instruction clears the channel d input word flag and transmits a resume signal over the input channel after the word reads into A.

71dm Input (A) Words to m from Channel d



This instruction reads a block of data arriving on input channel d and stores the data in consecutive address locations in storage. The initial storage location for the block is specified by the m designator. The length of the block is specified by the initial content of the A register or by a record flag on the input channel.

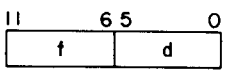
Instruction execution begins with a storage reference for the m designator. This quantity reads into the X register and then enters the Q register. Q then contains the address for the first word of the data block. The d designator specifies the channel number, and A contains a word count for the block. If the content of A is zero at this time, the instruction sequence terminates, and the next instruction word reads from storage.

The channel d input word flag must set before the first word of the block enters in storage. If this flag is not set when the instruction initiates, the PPU program stops with the instruction in the fd register and waits until the flag is set by an external signal. The presence of a channel d input record flag is ignored for the first word of the block.

When the channel d input word flag sets, the word on the input channel data lines reads into PPU storage at the location determined by the content of Q. The content of A reduces by one count. The content of Q increases by one count in a 12-bit one's complement mode. The channel d input word and record flags clear, and a resume pulse transmits over the input channel. If the content of A is zero, the instruction sequence terminates and the next instruction word reads from storage. If the content of A is not zero, the PPU program waits for the setting of the channel d input word flag for the next word of the block.

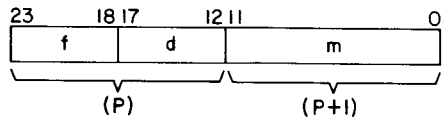
The setting of the channel d input record flag terminates the block input at any word after the first word. The sequence terminates with the content of A decremented by the number of words actually transmitted over the input channel. A noise word enters in the next sequential storage location in the PPU block input storage area. The remaining locations in the PPU storage area are unaltered.

72d Output from A on Channel d



This instruction transmits one word over output channel d from the lowest-order 12 bits of the content of the A register; the content is not altered in the process. This instruction does not execute while the channel d output word flag is set. If the flag is set from a previous output instruction, the PPU program stops with this instruction in the fd register and waits for an external resume signal to clear the channel d output word flag. When this instruction executes, the output word flag sets, and a word pulse transmits over output channel d.

73dm Output (A) Words from m on Channel d



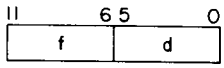
This instruction transmits a block of data over output channel d from consecutive storage locations beginning at address m. The length of the block is specified by the initial contents of the A register. A zero length causes the instruction to execute as a pass instruction.

Instruction execution begins with a storage reference for the m designator. This quantity reads into the X register and then enters the Q register. Q then contains the address for the first word of the data block. The d designator specifies the channel number, and A contains the word count for the block. If the content of A is zero at this time, the instruction sequence terminates, and the next instruction word reads from storage.

The channel d output word flag must clear before the first word of the block transmits over the channel. If this flag sets when the instruction initiates, the PPU program stops with the instruction in the fd register and waits until the flag clears by a resume pulse over output channel d. The presence of the channel d output record flag has no effect on the execution of this instruction.

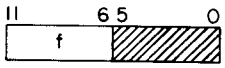
When the channel d output word flag clears, a word reads from storage location, determined by the content of Q, and enters the channel d output register. The channel d output word flag sets, and a word pulse transmits over the output channel. The content of A reduces by one count. The content of Q increases by one count in a 12-bit one's complement mode. If the content of A is zero, the instruction terminates, and the next instruction reads from storage. If the content of A is not zero, the PPU program waits for the channel d output word flag to clear and repeats the sequence for the next word of the block.

74d Set Output Record Flag on Channel d



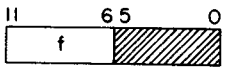
This instruction sets the channel d output record flag and transmits a record pulse over output channel d. The previous status of the flag is ignored in this process. The instruction executes and a record pulse transmits even though the channel d output record flag was already set.

75xx, 76xx Pass



These two instructions are identical and perform no logical function. Each instruction results in a 5-clock-period delay.

77xx Error Stop



This instruction causes the PPU program to stop and indicate a program error condition to the status and control register. The PPU can be restarted only by a deadstart.

PPU INSTRUCTION TIMING

Execution times for the PPU are listed in table 4-9. The timing notes refer to the notes at the end of the table. The clock periods, for execution times, are each 27.5 nanoseconds.

The execution timing for the PPU instructions is dominated by the access time of the core storage banks. There are two independent banks of storage. One bank contains all even storage addresses, and the other bank contains all odd storage addresses. If references to storage alternate between even and odd addresses, each reference requires 5 clock periods. If two even references (or two odd references) occur consecutively, the storage read/write cycle for the first reference must be completed before the second reference can begin. In this case, a storage reference requires 10 clock periods. As a result, the execution time for most of the PPU instructions is a multiple of clock periods with variation in increments of 5 clock periods, depending upon the storage addresses involved.

TABLE 4-9. PPU INSTRUCTION TIMING

Instruction Code	Description	Execution Time (Clock Periods)	Timing Notes
00xx	Error stop	-	-
0100m	Long jump to m	10 or 15	1, 2
01dm	Long jump to m + (d)	15, 20 or 25	1, 2
0200m	Return jump to m	15 or 20	1, 2
02dm	Return jump to m + (d)	20, 25 or 30	1, 2
03d	Unconditional jump d	7 or 10	2
04d	Zero jump d	5	3, 4
05d	Nonzero jump d	5	3
06d	Plus jump d	5	3
07d	Minus jump d	5	3
10d	Shift (A) by d	6	5
11d	Logical difference (A) and d	5	6
12d	Logical product (A) and d	5	6
13d	Selective clear (A) by d	5	6
14d	Load d	5	6
15d	Load complement d	5	6
16d	Add (A) + d	5	6
17d	Subtract (A) - d	5	6

TABLE 4-9. PPU INSTRUCTION TIMING (Contd)

Instruction Code	Description	Execution Time (Clock Periods)	Timing Notes
20dm	Load dm	10	1, 6
21dm	Add (A) + dm	10	1, 6
22dm	Logical product (A) and dm	10	1, 6
23dm	Logical difference (A) and dm	10	1, 6
24xx	Pass	5	6
25xx	Pass	5	6
26xx	Pass	5	6
27xx	Pass	5	6
30d	Load (d)	15	7
31d	Add (A) + (d)	15	7
32d	Subtract (A) - (d)	15	7
33d	Logical difference (A) and (d)	15	7
34d	Store (A) at (d)	15	7
35d	Replace add (A) + (d)	25	7
36d	Replace add one (d)	25	7
37d	Replace subtract one (d)	25	7
40d	Load ((d))	15 or 25	2
41d	Add (A) + ((d))	15 or 25	2
42d	Subtract (A) - ((d))	15 or 25	2
43d	Logical difference (A) and ((d))	15 or 25	2
44d	Store (A) at ((d))	15 or 25	2
45d	Replace add (A) + ((d))	25 or 35	2
46d	Replace add one ((d))	25 or 35	2
47d	Replace subtract one ((d))	25 or 35	2
5000m	Load (m)	20	1, 7
50dm	Load (m + (d))	20 or 30	1, 2
5100m	Add (A) + (m)	20	1, 7
51dm	Add (A) + (m+(d))	20 or 30	1, 2
5200m	Subtract (A) - (m)	20	1, 7
52dm	Subtract (A) - (m + (d))	20 or 30	1, 2
5300m	Logical difference (A) and (m)	20	1, 7
53dm	Logical difference (A) and (m+(d))	20 or 30	1, 2
5400m	Store (A) at (m)	20	1, 7
54dm	Store (A) at (m + (d))	20 or 30	1, 2
5500m	Replace add (A) + (m)	30	1, 7
55dm	Replace add (A) + (m + (d))	30 or 40	1, 2
5600m	Replace add one (m)	30	1, 7
56dm	Replace add one (m + (d))	30 or 40	1, 2
5700m	Replace subtract one (m)	30	1, 7
57dm	Replace subtract one (m + (d))	30 or 40	1, 2
60dm	Jump to m if channel d input word flag set	10	1, 8
61dm	Jump to m if channel d input word flag not set	10	1, 8
62dm	Jump to m if channel d input record flag set	10	1, 8

TABLE 4-9. PPU INSTRUCTION TIMING (Contd)

Instruction Code	Description	Execution Time (Clock Periods)	Timing Notes
63dm	Jump to m if channel d input record flag not set	10	1, 8
64dm	Jump to m if channel d output word flag set	10	1, 8
65dm	Jump to m if channel d output word flag not set	10	1, 8
66dm	Jump to m if channel d output record flag set	10	1, 8
67dm	Jump to m if channel d output record flag not set	10	1, 8
70d	Input to A from channel d	9	9
71dm	Input (A) words to m from channel d	24 or 42	1, 10
72d	Output from A on channel d	9	11
73dm	Output (A) words from m on channel d	34	1, 12
74d	Set output record flag on channel d	5	6
75xx	Pass	5	6
76xx	Pass	5	6
77xx	Error stop	-	-

Timing Notes:

1. Storage reference for second word of current instruction word must be to alternate bank.
2. Shorter time is obtained when full use is made of bank phasing (back-to-back storage references to alternate banks).
3. Time assumes that jump conditions are not met. If jump is met, time is same as for 03d instruction.
4. Designator d cannot by 00 or 77.
5. Time assumes that d equals three or less. Time increases by 1 clock period for each shift beyond three. Maximum time is 34 clock periods.
6. Storage reference(s) following the one for current instruction word must be to alternate bank(s).
7. Storage reference(s) following the one for current instruction word may be to either bank.
8. Time assumes that either jump conditions are not met or jump is taken to alternate bank. If jump is taken to same bank, time is 15 clock periods.
9. Time assumes that channel d input word flag is set. If not set, add time waiting for flag to set.
10. First time is for a two-word block input terminated by reducing the quantity in A to zero. Following assumptions are made.

    a. A count of 2 is in A.
    b. Channel d input word flag initially sets.
    c. First data storage reference is to alternate bank.
    d. Response time between resume pulse and setting of the input word flag is 2 clock periods.

    Second time is for a three-word block input terminated by setting the channel d input record flag. Following assumptions are made.

    a. Channel d input word flag initially sets.
    b. First data storage reference is to alternate bank.
    c. Response time between resume pulse and setting of the input word flag is 2 clock periods.

11. Time assumes that channel d output word flag is clear. If not clear, add time waiting for flag to clear.
12. Time is for a three-word block output. Following assumptions are made.

    a. A count of 3 is in A.
    b. Channel d output word flag initially clears.
    c. First data storage reference is to alternate bank.
    d. The device response time from receipt of word pulse to transmission of resume pulse is 2 clock periods.
    e. Assumes a 2-clock-period delay for word pulses and resume pulses between the PPU and the device.

PERIPHERAL PROCESSOR SUBSYSTEM INSTRUCTIONS — MODELS 171 THROUGH 176

PERIPHERAL PROCESSOR SUBSYSTEM INSTRUCTIONS – MODELS 171 THROUGH 176

Each PP sequentially executes instructions from its own memory and uses an 18-bit A register for manipulative operations. The A register is the only PP register used by the programmer. All the PPS arithmetic operations are binary and are performed in a one's complement mode. This mode treats a value of 777777 in A register as a negative zero.

The PPS instructions are the same and produce the same results as the PPU instructions except for the instructions listed in table 4-6. This table and other information for the instruction formats, designators, and addressing modes are located at the beginning of the previous subsection, Peripheral Processor Unit Instructions-Model 176.

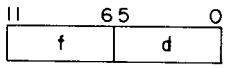
The following PPS instruction descriptions are briefly stated to avoid word-for-word repetitions of similar PPU instruction descriptions in the previous pages. Refer to corresponding PPU descriptions when additional instruction detail is required.

PPS INSTRUCTION DESCRIPTIONS

The PPS instructions have separate descriptions. Shaded areas, like those in the 260x and 261x instruction formats, indicate unused bits. The unused bits are ignored by the PPs.

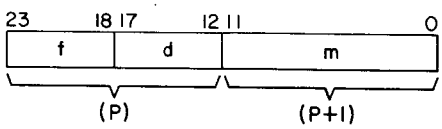
Timing information follows the instructions.

0000 Pass



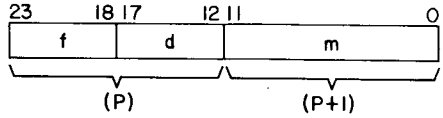
This 12-bit instruction specifies that no operation is to be performed. The instruction provides a means of padding out a program.

01dm Long Jump to m + (d)



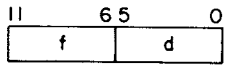
This 24-bit instruction jumps to the address given by m plus the content of location d. If d equals zero, m is not modified.

02dm Return Jump to m + (d)



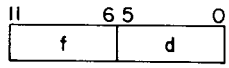
This 24-bit instruction jumps to the address given by m plus the content of location d. If d equals zero, m is not modified. The current program address (P) plus 2 is stored at the jump address. The next instruction starts at the jump address plus 1. The subprogram exits with a long jump, or normal sequencing to the jump address minus 1, which in turn contains a long jump, 0100. This returns the original program address plus 2 to the P register.

03d Unconditional Jump d



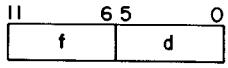
This 12-bit instruction provides an unconditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. The value of d is added to the current program address. If d is positive (01 through 37), 0001 through 0037 is added, and the jump is forward. If d 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



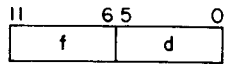
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the content of the A register is zero, the jump is taken. If the content of A is nonzero, the next instruction executes from P plus 1. Negative zero (777777) is treated as nonzero. For interpretation of d, refer to 03 instruction.

05d Nonzero Jump d



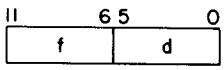
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the content of the A register is nonzero, the jump is taken. If the content of A is zero, the next instruction executes from P plus 1. Negative zero (777777) is treated as nonzero. For interpretation of d, refer to 03 instruction.

06d Plus Jump d



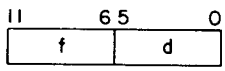
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the sign of the A register is positive, the jump is taken. If the sign of A is negative, the next instruction executes from P plus 1. Positve zero is treated as a positive quantity. Negative zero is treated as a negative quantity. For interpretation of d, refer to 03 instruction.

07d Minus Jump d



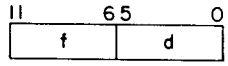
This 12-bit instruction provides a conditional jump to any address up to 31 (decimal) locations forward or backward from the current program address. If the content of the A register is negative, the jump is taken. If the content of A is positive, the next instruction is executed from P plus 1. Positive zero is treated as a positive quantity. Negative zero is treated as a negative quantity. For interpretation of d, refer to 03 instruction.

10d Shift d



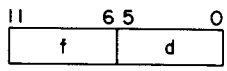
This 12-bit instruction shifts the content of the A register right or left d places. If d is positive (00 through 37), the shift is left circular. If d is negative (40 through 77), the shift is right (end-off with no sign extension). Thus, d equal to 06 requires a left shift of six places; d equal to 71 requires a right shift of six places.

11d Logical Difference d



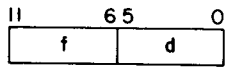
This 12-bit instruction forms the bit-by-bit logical difference of d and the lower six bits of A in the register in A. This is equivalent to complementing individual bits of A that correspond to bits of d that are one. The upper 12 bits of A are not altered.

12d Logical Product d



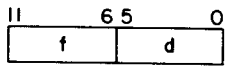
This 12-bit instruction forms the bit-by-bit logical product of d and the lower six bits of the A register and leaves this quantity in the lower six bits of A. The upper 12 bits of A are zero.

13d Selective Clear d



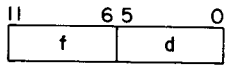
This 12-bit instruction clears any of the lower six bits of the A register where corresponding bits of d are one. The upper 12 bits of A are not altered.

14d Load d



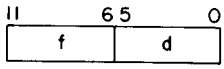
This 12-bit instruction clears the A register and loads d. The upper 12 bits of A are zero.

15d Load Complement d



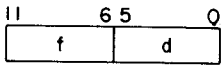
This 12-bit instruction clears the A register and loads the complement of d. The upper 12 bits of A are one.

16d Add d



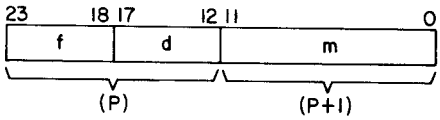
This 12-bit instruction adds d (treated as a 6-bit positive quantity) to the content of the A register.

17d Subtract d



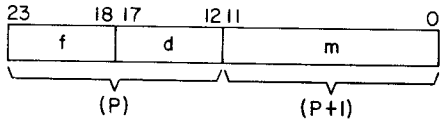
This 12-bit instruction subtracts d (treated as a 6-bit positive quantity) from the content of the A register.

20dm Load dm



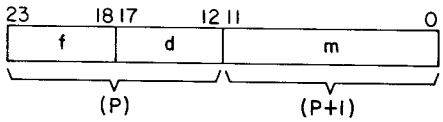
This 24-bit instruction clears the A register and loads an 18-bit quantity consisting of d as the upper 6 bits and m as the lower 12 bits. The content of the location (P+1) which follows the present program address (P) is read to provide m.

21dm Add dm



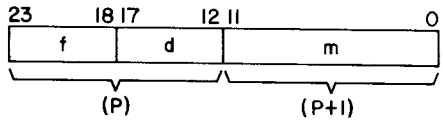
This 24-bit instruction adds the 18-bit quantity consisting of d as the upper 6 bits and m as the lower 12 bits to the A register. The content of the location (P+1) which follows the present program address (P) is read to provide m.

22dm Logical Product dm



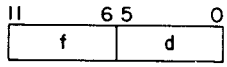
This 24-bit instruction forms the bit-by-bit logical product of the content of the A register, and the 18-bit quantity dm in A. The upper 6 bits of this quantity consist of d, and the lower 12 bits are the content of the location (P+1) which follows the present program address (P).

23dm Logical Difference dm



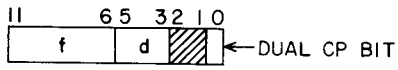
This 24-bit instruction forms the bit-by-bit logical difference of the content of the A register and the 18-bit quantity dm in A. This is equivalent to complementing individual bits of A which correspond to bits of dm that are one. The upper 6 bits of the quantity consist of d, and the lower 12 bits are the content of the location (P+1) which follows the present program address (P).

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 — Models 171 through 175



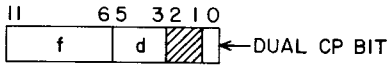
This 12-bit instruction transmits an 18-bit, absolute address from the A register to the CP with a signal which tells the CP to perform an exchange jump. The address in A is the starting location of an exchange package of 16 words containing information for a CP program to be executed. The 18-bit initial 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 dual-CP systems, the lowest-order bit of the instruction format specifies which of the two CPs the exchange jump interrupts. In single-CP systems, this bit is not interpreted.

260x Exchange Jump — Model 176

This instruction performs as a 261x instruction.

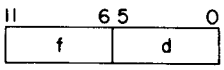
261x Monitor Exchange Jump — Models 171 through 175



This 12-bit instruction is enabled or disabled by the CEJ/MEJ switch. When the switch is in the ENABLE position, this instruction causes a conditional exchange jump of the CP. If the monitor flag is clear, this instruction initiates the exchange jump and sets the flag. If the monitor flag is set, 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 exchange address is an absolute address. If the CEJ/MEJ switch is in the DISABLE position, this instruction performs as a 260 instruction.

In dual-CP systems, the lowest-order bit of the instruction format specifies which of the two CPs the exchange jump interrupts. In single-CP systems, this bit is not interpreted.

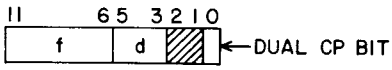
261x Monitor Exchange Jump — Model 176



This 12-bit instruction is not controlled by the CEJ/MEJ switch on the deadstart panel. The CEJ/MEJ switch has no function in Model 176.

If the monitor mode flag clears and no I/O interrupts are waiting to be processed, the instruction initiates an exchange jump of the CP to the 18-bit address specified by the A register. If the monitor mode flag sets or I/O interrupts are waiting to be processed, this instruction acts as a pass instruction.

262x Monitor Exchange Jump to MA — Models 171 through 175



This 12-bit instruction is enabled or disabled by the CEJ/MEJ switch. When the switch is in the ENABLE position, this instruction causes a conditional exchange jump of the CP. If the monitor flag is clear, this instruction initiates the exchange jump and sets the flag. If the monitor flag is set, 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.

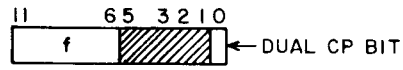
This exchange address is an absolute address. If the CEJ/MEJ switch is in the DISABLE position, this instruction performs as a 260 instruction.

In dual-CP systems, the lowest-order bit of the instruction format specifies which of the two CPs the exchange jump interrupts. In single-CP systems, this bit is not interpreted.

262x Monitor Exchange Jump to MA — Model 176

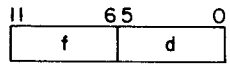
This instruction performs as a 261x instruction.

27x Read Program Address



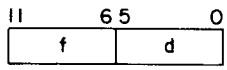
This 12-bit instruction transfers the content of the CP P register to the PP A register; this allows the PP to determine whether the CP is running. For information on the dual-CP bit, refer to the 260x instruction.

30d Load (d)



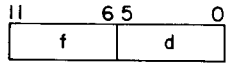
This 12-bit instruction clears the A register and loads the content of location d. The upper six bits of A are zero.

31d Add (d)



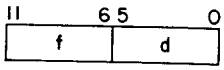
This 12-bit instruction adds the content of location d (treated as a 12-bit positive quantity) to the A register.

32d Subtract (d)



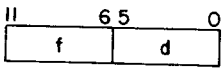
This 12-bit instruction subtracts the content of location d (treated as a 12-bit positive quantity) from the A register.

33d Logical Difference (d)



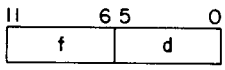
This 12-bit instruction forms the bit-by-bit logical difference of the lower 12 bits of the A register and the content of location d in the A register. This is equivalent to complementing individual bits of A which correspond to bits in location d that are ones. The upper six bits are not altered.

34d Store d



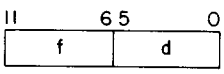
This 12-bit instruction stores the lower 12 bits of the A register in location d.

35d Replace Add (d)



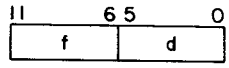
This 12-bit instruction adds the quantity in location d to the content of the A register and stores the lower 12 bits of the result in location d. The result remains in A at the end of the operation, and the original content of A is destroyed.

36d Replace Add One (d)



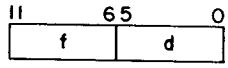
This 12-bit instruction replaces the quantity in location d with its original value plus 1. The result remains in the A register at the end of the operation, and the original content of A is destroyed.

37d Replace Subtract One (d)



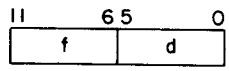
This 12-bit instruction replaces the quantity in location d with its original value minus 1. The result remains in the A register at the end of the operation, and the original content of A is destroyed.

40d Load ((d))



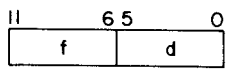
This 12-bit instruction clears the A register and loads a 12-bit quantity that is obtained by indirect addressing. The upper six bits of A are zero. Location d is read from PPM, and the word read is used as the operand address.

41d Add ((d))



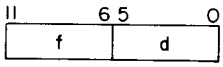
This 12-bit instruction adds a 12-bit operand (treated as a positive quantity) obtained by indirect addressing to the content of the A register. Location d is read from PPM, and the word read is used as the operand address.

42d Subtract ((d))



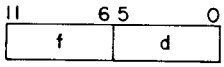
This 12-bit instruction subtracts a 12-bit operand (treated as a positive quantity) obtained by indirect addressing from the A register. Location d is read from PPM, and the word read is used as the operand address.

43d Logical Difference ((d))



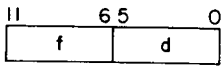
This 12-bit instruction forms the bit-by-bit logical difference of the lower 12 bits of the A register and the 12-bit operand read by indirect addressing in the A register. Location d is read from PPM, and the word read is used as the operand address. The upper six bits of A are not altered.

44d Store ((d))



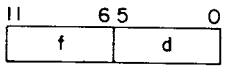
This 12-bit instruction stores the lower 12 bits of the A register in the location specified by the content of location d.

45d Replace Add ((d))



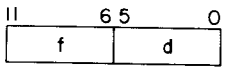
This 12-bit instruction adds the operand, which is obtained from the location specified by the content of location d, to the content of the A register. The lower 12 bits of the sum replace the original operand. The result remains in A at the end of the operation.

46d Replace Add One ((d))



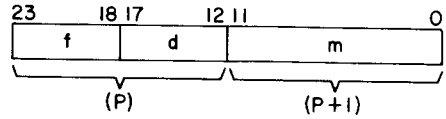
This 12-bit instruction replaces the operand, which is obtained from the location specified by the content of location d, by its original value plus 1. The result remains in the A register at the end of the operation, and the original content of A is destroyed.

47d Replace Subtract One ((d))



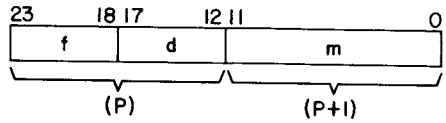
This 12-bit instruction replaces the operand, which is obtained from the location specified by the content of location d, by its original value minus 1. The result remains in the A register at the end of the operation, and the original content of A is destroyed.

50dm Load (m + (d))



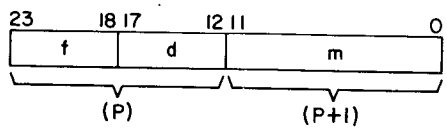
This 24-bit instruction clears the A register and loads a 12-bit quantity. The upper six bits of A are zeros. The 12-bit operand is obtained by indexed direct addressing. The quantity m, read from PPM location P plus 1, serves as the base operand address to which the content of d is added. If d equals 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))



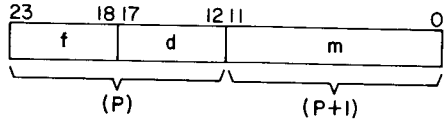
This 24-bit instruction adds the 12-bit operand (treated as a positive quantity) read by indexed direct addressing (refer to 50 instruction) to the A register.

52dm Subtract (m + (d))



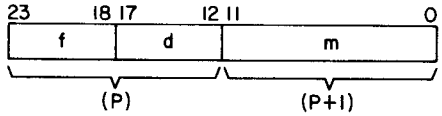
This 24-bit instruction subtracts the 12-bit operand (treated as a positive quantity) read by indexed direct addressing (refer to 50 instruction) from the A register.

53dm Logical Difference (m + (d))



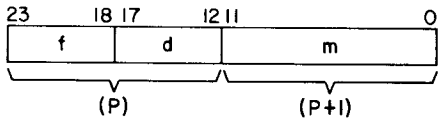
This 24-bit instruction forms the bit-by-bit logical difference of the lower 12 bits of the A register and a 12-bit operand obtained by indexed direct addressing in A. The upper six bits of A are not altered.

54dm Store (m + (d))



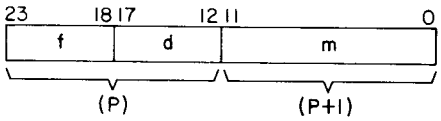
This 24-bit instruction stores the lower 12 bits of the A register in the location determined by indexed addressing (refer to 50 instruction).

55dm Replace Add (m + (d))



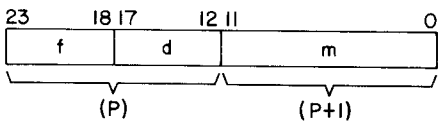
This 24-bit instruction adds the operand, which is obtained from the location determined by indexed direct addressing, to the A register. The lower 12 bits of the sum replace the original operand in PPM. The result remains in A at the end of the operation, and the original content of A is destroyed.

56dm Replace Add One (m + (d))



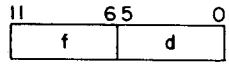
This 24-bit instruction replaces the operand, which is obtained from the location determined by indexed direct addressing, by its original value plus 1 (refer to 50 instruction). The result remains in the A register at the end of the operation, and the original content of A is destroyed.

57dm Replace Subtract One (m + (d))



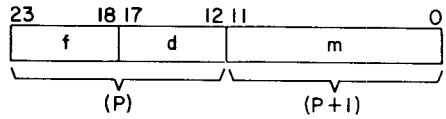
This 24-bit instruction replaces the operand, which is obtained from the location determined by indexed direct addressing, by its original value minus 1 (refer to 50 instruction). The result remains in the A register at the end of the operation, and the original content of A is destroyed.

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

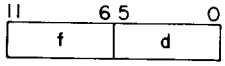


This 24-bit instruction reads a block of 60-bit words from CM. Location d contains the block length. An 18-bit address of the first central word must be loaded into the A register prior to executing this instruction. (This is an absolute address.) During the execution of the instruction, the content 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 reduces by one as each central word processes. The content of address 0 increments by one and enters the P register at the end of the instruction.

Each central word disassembles into five 12-bit words beginning with the highest-order 12 bits. The first word stores at PPM location m. The content of P (which is holding m) advances by one to provide the next address in the PPM as each 12-bit word is stored. If P overflows, operation continues as P advances 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 advances by one to provide the next CM address after each 60-bit word is disassembled and stored. The content of the Q register also reduces by one. The block transfer completes 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.

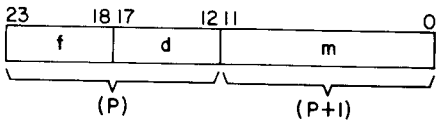
62d Central Write to (A) from d



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



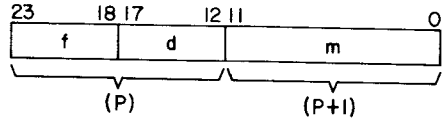
This 24-bit instruction assembles a block of 60-bit words and writes them in CM. Location d holds the number of 60-bit words. The A register holds the beginning CM address. (This is an absolute address.) During the execution of this instruction, the content of P (P plus 1) goes to PP address 0, and m enters the P register. The content of d enters the Q register, where it reduces by one as each central word is assembled. The content of address 0 increments by one and enters 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 PPM. This word appears as the highest-order 12 bits of the first 60-bit word to be stored in CM.

P advances by one to provide the next address in PPM as each 12-bit word is read. If P overflows, operation continues as P advances 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.

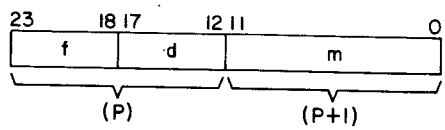
A advances by one to provide the next CM address after each 60-bit word is assembled. Q also reduces by one. The block transfer completes when Q equals zero.

64dm Jump to m if Channel d Active



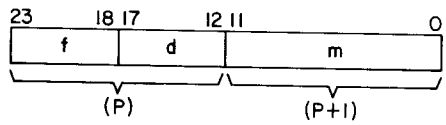
This 24-bit instruction provides a conditional jump to a new address specified by m. The jump is taken if the channel specified by d is active. The next instruction is at P plus 2 if the channel is inactive.

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.

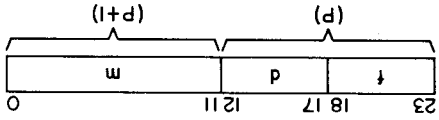
66dm Jump to m if Channel d Full



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.

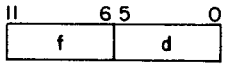
An input channel is full when the input equipment places a word in the channel and that word has not been accepted by a PP. The channel is empty when a word has been accepted. An output channel is full when a PP places a word on the channel. The channel is empty when the output equipment accepts the 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 66 instruction for explanation of full and empty.)

70d Input to A from Channel d

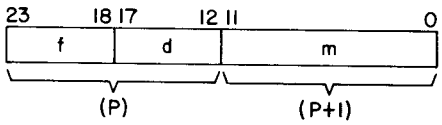


This 12-bit instruction transfers a word from input channel d to the lower 12 bits of the A register. The upper six bits of A are cleared to zero.

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



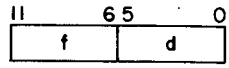
This 24-bit instruction transfers a block of 12-bit words from input channel d to PPM. The first word goes to the PPM address specified by m. The A register holds the block length. The content of A reduces by one as each word is read. The input operation completes when A equals zero or the data channel becomes inactive. If the operation terminates by the channel becoming inactive, the next storage location in PPM is set to zero. However, the word count is not affected by this empty word. Therefore, A 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 or Channel d

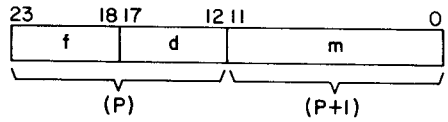


This 12-bit instruction transfers a word from the A register (lower 12 bits) to output channel d.

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



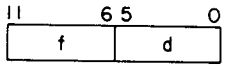
This 24-bit instruction transfers a block of words from PPM to channel d. The first word is read from the address specified by m. The A register holds the number of words to be sent. A reduces by one as each word is read. The output operation completes when A equals zero or the channel becomes inactive.

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 executes when the data channel is inactive, no output operation is accomplished, and the program continues at P plus 2.

74d Activate Channel d

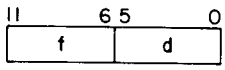


This 12-bit instruction activates the channel specified by d and sends the active signal on the channel to equipment connected to the channel. Activating a channel, which must precede a 70 through 73 instruction, prepares I/O equipment for the exchange of data.

NOTE

If this instruction executes 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.

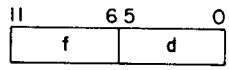
NOTE

If this instruction executes when the data channel is already inactive and bit 5 of d is set, the program continues at P plus 1. The channel remains inactive, and no inactive signal is sent to the I/O equipment. Deactivating an already inactive channel causes the PP to hang until the channel becomes active.

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.

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

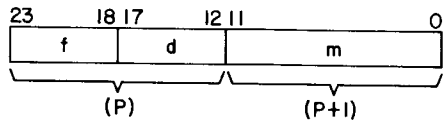


This 12-bit instruction sends the external function code in the lower 12 bits of the A register on channel d.

NOTE

If this instruction executes 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 transmit. The channel remains active, and execution continues.

77dm Function m on Channel d



This 24-bit instruction sends the external function code specified by m on channel d.

NOTE

If this instruction executes 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 transmits. The channel remains active, and execution continues.

PPS INSTRUCTION TIMING

Execution times for the PPS instructions are listed in table 4-10. The times listed in the execution time column assume that no conflicts occur. The timing notes refer to the notes 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 4-10. PPS INSTRUCTION TIMING

Instruction Code	Description	Execution Time (Minor Cycles)	Timing Notes
0000	Pass	10	1
01dm	Long jump to m + (d)	-	2
02dm	Return jump to m + (d)	-	
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	10	3
261x	Monitor exchange jump	10	3
262x	Monitor exchange jump to MA	10	3
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	
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, 6
61dm	Central read (d) words from (A) to m	-	7, 8
62d	Central write to (A) from d	60	8
63dm	Central write (d) words to (A) from m	-	8, 9
64dm	Jump to m if channel d active	20	

TABLE 4-10. PP INSTRUCTION TIMING (Contd)

Instruction Code	Description	Execution Time (Minor Cycles)	Timing Notes
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	10
71dm	Input (A) words to m from channel d	-	10, 11
72d	Output from A on channel d	20	10
73dm	Output (A) words from m on channel d	-	10, 11
74d	Activate channel d	20	10
75d	Disconnect channel d	20	10
76d	Function (A) on channel d	20	10
77dm	Function m on channel d	20	10

Timing Notes:

1. 30 cycles; if d = 0, 20 cycles.
2. 40 cycles; if d = 0, 30 cycles.
3. Assuming no CMC access conflicts.
4. 50 cycles; if d = 0, 40 cycles.
5. 60 cycles or 70 cycles if PPM refresh occurs on trip 5 of the PP around the barrel, at 1X speed. 80 cycles at 2X speed. (Refresh applies to models A and B only.)
6. Assuming no CMC conflicts and no CSU refresh. (Refresh applies to models A and B only.)
7. 60 cycles plus 50 cycles per word at 1X speed. 60 cycles plus 60 cycles per word at 2X speed.
8. Assuming no conflicts within CMC or pyramids.
9. 60 cycles plus 50 cycles per word.
10. Assuming no conflicts for an expanded PP system. (Refer to Channel Conflicts in An Expanded PP System in Section 5.)
11. 50 cycles plus 10 cycles per word.

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 error responses. Unless otherwise specified, all information in this section is applicable to all models.

Refer to appendix B for specific model 175 and model 176 differences.

CENTRAL PROCESSOR PROGRAMMING

CENTRAL PROCESSOR PROGRAMMING

The central processor (CP) uses an exchange jump operation to switch programs. The execution of an exchange jump permits the CP to send pertinent information from the operating and control registers to central memory (CM) and permits CM to send new information to the same registers. The information that flows out of and into the operating and control registers during an exchange jump is called an exchange package. The exchange package differs between models 171 through 175 and model 176.

EXCHANGE JUMP — MODELS 171 THROUGH 175

An exchange jump instruction is a 013 in the CP and 260, 261, or 262 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 CM. The address is K plus the content of the Bj register or the monitor address for the CP-initiated exchange. The address is the content of the A register of PPS-0 or PPS-1 or the content of the monitor address (MA) register in the PPS-initiated exchange. The PPS also has the monitor exchange jump to MA, 262, instruction in which the content of MA is used for the exchange address. The exchange package (figure 5-1) provides the following information for 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 (lower six bits are assumed to be zeros)

Field length of block transfer for extended core storage (FLE) - 24 bits (lower six bits are assumed to be zeros)

Monitor address - 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 contains constant 0) registers - 18 bits

The time that 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.

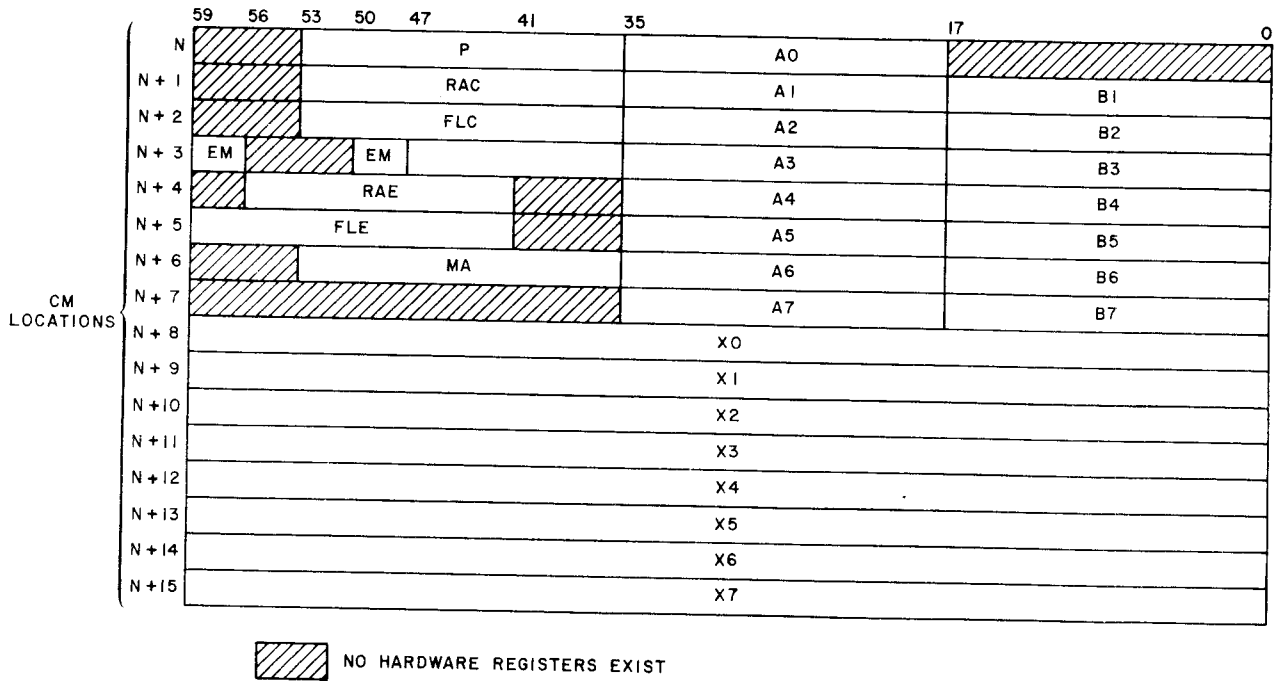


Figure 5-1. Exchange Package - Models 171 through 175

When the flag is set, the CP is in a noninterruptible monitor mode. When the flag is clear, the CP is in an interruptible program mode. A master clear (deadstart) clears the MF.

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 013 instruction

PP 260x, 261x, and 262x instructions

The central exchange jump/monitor exchange jump (CEJ/MEJ) switch 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
260x, 261x, and 262x instructions	Exchange jump to the address in A of the CPU selected by x. When x is 0, CPU-0 is selected. When x is 1, CPU-1 is selected. If x is 1 and CPU-1 is not present, the exchange jump is to CPU-0.

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 of the exchange package is K plus the content of Bj, and MF clears.
260x instruction	Exchange jump to the address in A.
261x 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.
262x 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.

EXCHANGE JUMP — MODEL 176

An exchange jump instruction is 013 in the CP and 26 in the PPS. The instruction interrupts the CP and provides CM with the first address of a 16-word exchange package. The address is K plus the content of the Bj register plus the reference address for CM or the normal exit address. The address is the content of the A register of PPS-0 or PPS-1 in the PPS-initiated (026) exchange. The exchange package (figure 5-2) provides the following information for a program to be executed.

Program address (P) - 18 bits

Reference address for CM (RAS) - 18 bits

Field length of program for CM (FLS) - 18 bits

Reference address for LCME (RAL) - 22 bits

Field length of program for LCME (FLL) - 22 bits

Program status designator register (PSD) - 18 bits

Normal exit address (NEA) - 18 bits

Error exit address (EEA) - 18 bits

NOTE

Bit 53 of word N+7 is used as a flag and is not used as bit 17 of the EEA. For a description of its use, refer to the description of the multiplexer (MUX).

Current contents of eight A registers

Current contents of eight X registers

Current contents of B1 through B7 registers

The period of time during which a particular exchange package resides in the central processor unit (CPU) hardware registers is termed the execution interval. The execution interval begins with an exchange jump that reads the exchange package from CM and enters these parameters into the CPU registers. It ends with another exchange jump that stores the exchange package back into CM.

Several instructions or conditions initiate exchange jumps and select the exchange package that is to begin execution.

Exchange exit instructions (013xx or 013jK)

Error exit

I/O interrupt

Real-time interrupt

Step mode

Exchange Exit Instructions

The normal termination for an exchange package execution interval is caused by an exchange exit instruction (013xx or 013jK) in the associated program. The EM flag in the PSD register determines the source of the exchange package.

The EM flag indicates a privileged monitor program and is normally not set for an object program execution interval. When the flag is not set and the object program terminates the execution interval with an 013xx instruction, the NEA is the absolute address of the exchange package. When this flag is set and the program terminates the execution interval with an 013jK instruction, the absolute CM address for the exchange package forms by adding the content of Bj plus K.

Error Exit

An object program terminates with an exchange jump to the EEA register upon encountering an error exit instruction (00) or under certain conditions defined by the PSD register. Some of these conditions may be selected by the programmer and some are unconditional. In general, errors caused by arithmetic overflow, underflow, or indefinite results during computation may be allowed to proceed through the calculation or may cause an error exit, depending upon mode selection. Errors caused by hardware failure or program addressing from an assigned field in storage cause unconditional error exits. In any error exit case, the programmer may allow the object program to continue where the error can be corrected or ignored.

The error condition flags and mode selection flags are all contained in the PSD register, which is loaded from the exchange package for each program execution interval. The mode selections are made in the exchange package prior to the execution interval of the program. If an error condition occurs during the execution interval, the type of error can be determinted by analyzing the terminating

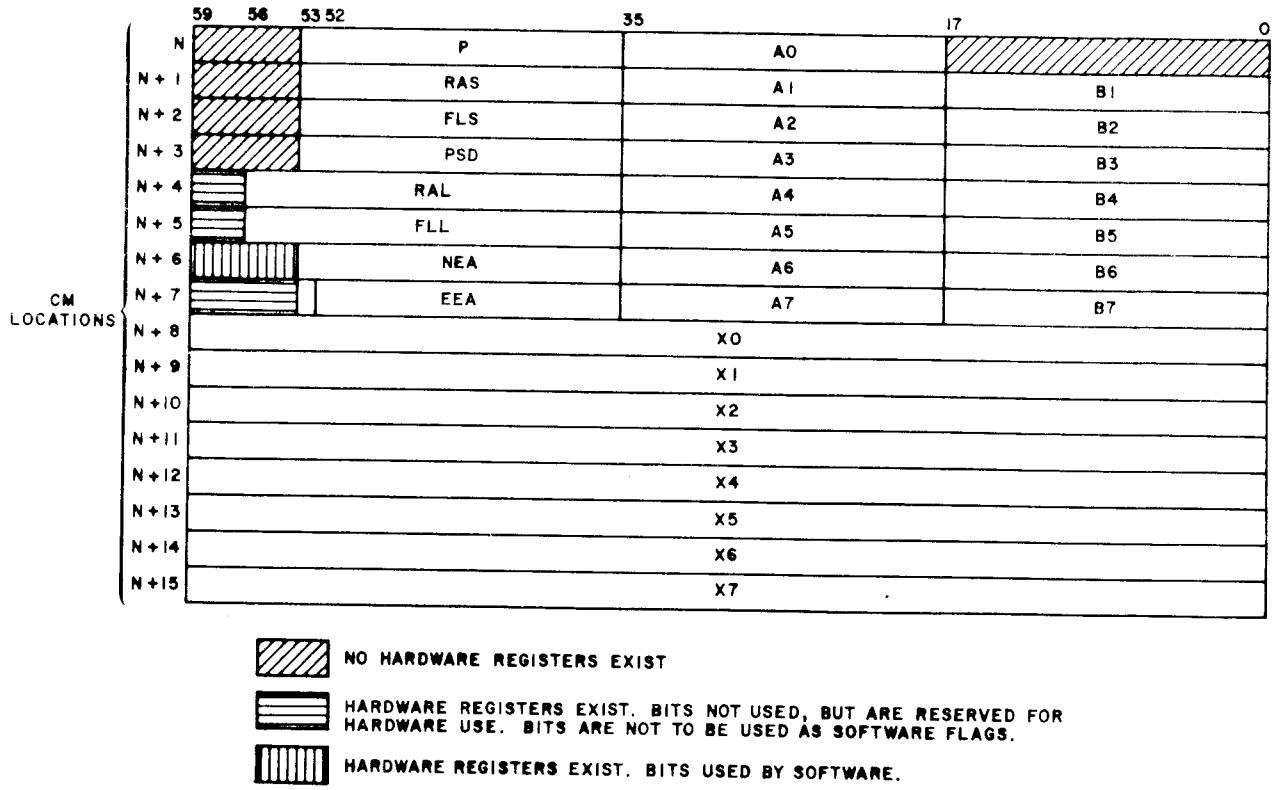


Figure 5-2. Exchange Package - Model 176

exchange package parameters. Each bit in the PSD register has significance either as a mode selection or an error condition flag.

Input/Output Interrupt

The MUX section of the CP monitors input/output (I/O) activity between the PPU and CM. The MUX issues an interrupt request to the CPU when the threshold of an CM input or output buffer is reached. A record pulse from a PPU also causes an interrupt request. When accepted, an I/O interrupt request initiates an exchange jump to the CPU program. An exchange request from the PPs also causes an interrupt request.

Real-Time Interrupt

Programs may be timed precisely by using the CPU clock period counter which advances one count each 27.5-nanosecond clock period. Since the clock advances synchronously with program execution, a program may be timed to an exact number of clock periods.

The CPU clock period counter contains a 17-bit register that can be sensed by a read input channel (0) status instruction. An overflow of the highest-order bit in this counter sets the real-time clock interrupt flag, which is actually the 18th bit of the register.

The real-time clock interrupt flag attempts an interrupt of the program to absolute address 0020 in CM each 3.6 milliseconds (approximate). The program to absolute address 0020 may change because of buffer bias bits. The real-time exchange package at this CM address executes a program that performs operations associated with the clock.

Step Mode

A program may be executed in step mode by setting the step mode flag in the PSD register for the program execution interval. Step mode causes the program to be interrupted at the end of each program instruction word with an exchange jump to EEA.

OPERATING CHARACTERISTICS — MODELS 171, 172, OR 174 WITH TWO CPs

Two CPs provide the following unique programming characteristics.

- When one CP is in monitor mode, a monitor exchange jump to either CP aborts. Since the exchange never starts, the instruction is a pass.
- When one CP is in monitor mode, a central exchange jump from the second CP hangs until the monitor flag clears in the first CP.
- If a regular exchange jump (2600) executes with a CEJ/MEJ instruction, the jump may cause the setting of both monitor flags. This condition can cause both CPs to hang on CEJ instructions.
- An ECS transfer in progress blocks a central exchange jump from either CP.
- A monitor exchange jump to a CP that has an extended core storage (ECS) transfer in progress is allowable. However, a monitor exchange jump to the other CP waits until the first CP completes the ECS transfer and then executes if neither monitor flag is set.
- A normal exchange jump to a CP doing an ECS transfer causes the CP to terminate the ECS transfer and execute the exchange jump. A normal exchange jump to the other CP is withheld until the ECS transfer is finished in the first CP.

OPERATING CHARACTERISTICS — MODEL 176

- When the monitor mode flag sets in the PSD register, interrupt requests, I/O interrupt requests, or peripheral processor subsystem (PPS) exchange requests are not honored. When the monitor mode flag clears, all interrupt requests are honored (in priority order).
- I/O channel interrupt exchange packages must have the monitor mode flag set in the PSD register. If this bit is not set, the I/O channel interrupt request causes repeated interrupts of the interrupt program.
- The CPU deadstart exchange jump is the result of the CPU deadstart (master clear) signal clearing the entire I/O channel interrupt request register. This results in a channel 0 interrupt request which causes an exchange jump when the CPU deadstart signal drops, using the exchange package for channel 0. Because the CPU deadstart exchange jump is the result of an I/O interrupt request, the deadstart exchange package must have the monitor mode flag set in the PSD register. If this bit is not set, the CPU deadstart program is reinterrupted by the channel 0 interrupt request.
- Like other exchange jump sequences, the CPU deadstart exchange jump swaps register data with CM exchange package data (locations 0 through 17). This exchange package (locations 0 through 17) can be relocated by the buffer bias bits. All exchange data swapped into CM is as it was in the CPU registers except for the PSD register data. The PSD bits are correct except for the unconditional clearing of the monitor mode flag and the unconditional setting of the program range flag. The program range flag sets because of the

time delay between the dropping of the CPU deadstart signal and the setting of the request interrupt flag (RIF).

- Six P registers are in the CPU hardware, each feeding different circuits. All P registers always contain the same value. Ensure that all P registers contain the same value when working on P-related problems.

- The 00 instruction can be blocked from setting the program range flag under the following condition.

  If an I/O interrupt request sets the RIF at the same time as the 00 instruction enters the translation bits of the current instruction word (CIW) top bits, the setting of the program range flag is blocked by RIF. The P register has advanced to the next location. If the next location contains legal instruction code, the I/O interrupt program returns control to this instruction word, and the 00 instruction is missed because the program range flag did not set.

- A master clear of the CPU can cause a CM parity error. To prevent a parity error caused by a master clear from being confused with a parity error caused by a system failure, check CM after each master clear to verify that it is free of parity errors. Verify the existence of any parity errors by reading all addresses. Eliminate any parity errors by writing into the affected addresses.

INSTRUCTION EXECUTION — MODELS 171 THROUGH 174

The model 171 through 174 CPs sequentially read and execute program instruction words from their CMs. The CPs read the instruction words with read next instruction (RNI) operations. These operations begin with an RNI initiation which occurs between executions of the first and second instruction in the program instruction word (figure 5-3) being processed. An RNI memory reference takes place in the remaining part of the RNI operation and occurs during the execution of the instructions that follow the RNI initiation. In case of a memory conflict between instructions and the RNI initiation, CMC delays the instructions until memory is not busy.

Calculation of the best-case execution time of a program instruction word requires adding the RNI initiation time to the total instruction execution times within the word. If the instruction that follows the RNI initiation does not require a memory reference, the RNI initiation time is 2 clock periods. If the instruction (such as a jump, branch, load, or store) that follows the RNI initiation does require a memory reference and there are no CM conflicts for that reference, the RNI initiation time is 5 clock periods. If the instruction has a CM conflict, the number of conflicts determines how many more additional clock periods for RNI are necessary.

Exceptions in the calculation of best-case program instruction word execution times occur with the jump or branch instructions. These instructions do not require the addition of the RNI initiation time if they occupy the upper position (parcel 0) of the program instruction word and their jump conditions

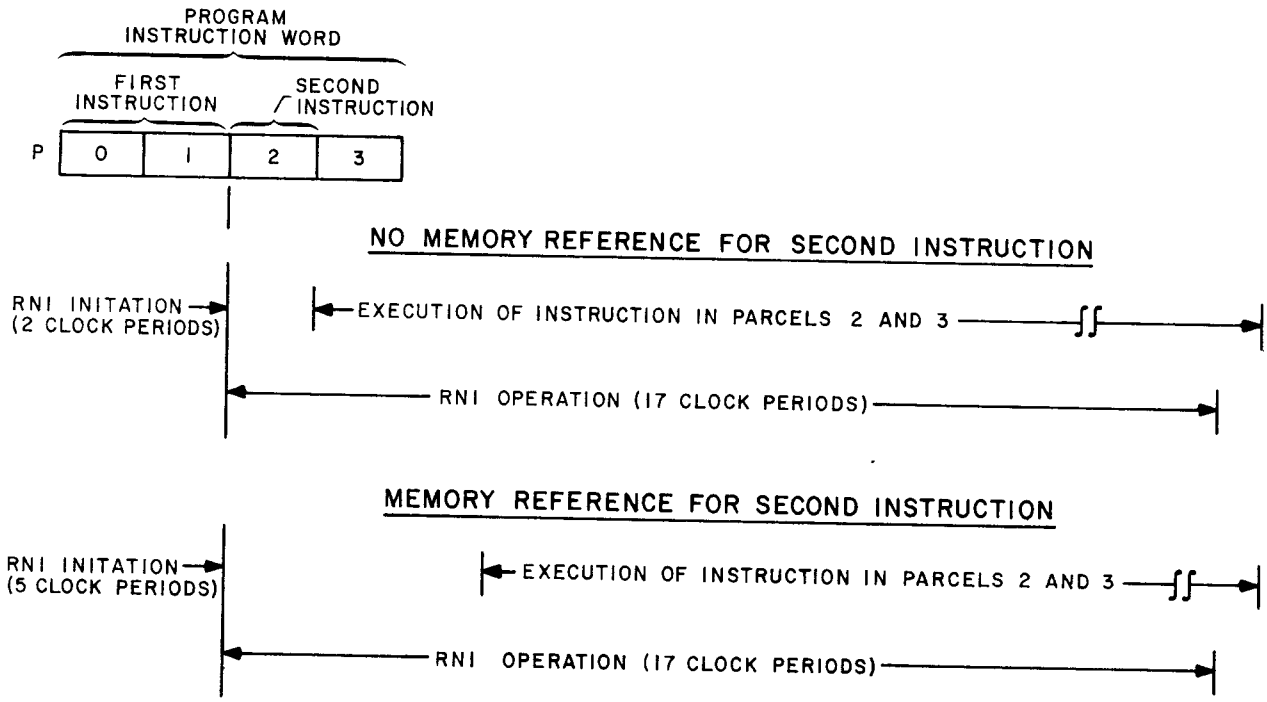
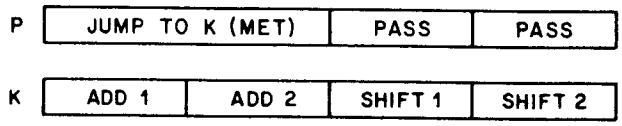


Figure 5-3. Instruction Execution - Models 171 Through 174

are met as shown in the following example. The exceptions occur because the execution times for the jump or branch instructions include the time required to read the new program instruction word at the jump or branch address.

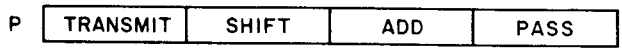


Instruction	Model 173 Clock Periods Required
Jump (met)	22
Add 1	6
RNI initiation	2
RNI completion (15 clock periods)	-
Add 2	6
Shift 1	6
Shift 2	6
Total	48

If the conditions for the jump or branch instruction are not met, the RNI initiation and RNI completion times must be added to the instruction time as shown in the following example.

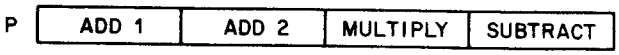
Instruction	Model 173 Clock Periods Required
Jump (not met)	5
RNI initiation	2
RNI completion	15
Pass 1 (3 clock periods)	-
Pass 2 (3 clock periods)	-
Total	22

The minimum time for execution of a program instruction word is the execution time of the first instruction in the word plus a minimum of 17 clock periods for the RNI operation. The following example shows that the instruction times that follow the RNI initiation are not part of the total clock period calculations if the instructions execute in less time than the time required for the RNI completion.



Instruction	Model 172 Clock Periods Required
Transmit	4
RNI initiation	2
RNI completion	15
Shift (6 clock periods)	-
Pass (3 clock periods)	-
Pass (3 clock periods)	-
Total	21

The maximum time for program instruction word completion is the execution time of the first instruction word plus the RNI initiation time plus the time required to complete the following instructions in the word. The following example shows that the time for RNI completion is not part of the total clock-period calculations when it is less than the time required for the execution of the instructions that follow the RNI initiation.



Instruction	Model 172 Clock Periods Required
Add 1	6
RNI initiation	2
RNI completion (15 clock periods)	-
Add 2	6
Shift	6
Subtract	6
Total	26

For program optimization in the CP, instructions requiring a memory reference must be in the upper part of the program instruction words. This optimization shortens the RNI initiation times from 5 to 2 clock periods if the instruction that follows the RNI initiation requires a memory reference. The optimization also prevents wait time that occurs when an unnecessary RNI operation occurs before a jump or branch instruction.

INSTRUCTION EXECUTION — MODELS 175 AND 176

Program instructions words read one at a time from the instruction word stack (IWS) into the 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 issues, it must complete 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 in model 175. If two-parcel instructions are initiated in parcel 3 on model 176, the lower parcel is all zeros.

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 alters to the new program address, and the corresponding word reads 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. In model 175, the stack is voided. Instruction execution continues upon receipt of the words from CM.

A branch from 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. In model 175, 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. In model 175, the IWS is 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. In model 176, the stack is voided by executing a return jump (01) instruction.

FLOATING-POINT ARITHMETIC — MODELS 171 THROUGH 176

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 5-4), 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 one's-complement notation.

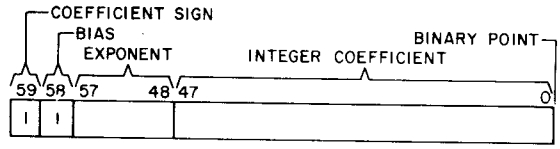


Figure 5-4. Floating-Point Format

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). Example 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.

The packed negative zero exponent is not used for normal operation. Instead, 1777 is used to indicate the special error condition of indefinite.

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 4771 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 generates 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, and it is the same as a zero word in integer format.

Indefinite

An indefinite result indicator generates 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 always generate 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)

Xj \ Xk	W	+∞	-∞	± IND
W		+∞	-∞	IND
+∞	+∞	+∞	IND	IND
-∞	-∞	IND	-∞	IND
± IND	IND	IND	IND	IND

TABLE 5-3. Xj MINUS Xk (31, 33, 35 INSTRUCTIONS)

Xj \ Xk	W	+∞	-∞	± IND
W		-∞	+∞	IND
+∞	+∞	IND	+∞	IND
-∞	-∞	-∞	IND	IND
± IND	IND	IND	IND	IND

TABLE 5-4. Xj MULTIPLIED BY Xk (40, 41, 42 INSTRUCTIONS)

Xj \ Xk	+N	-N	+0	-0	+∞	-∞	± IND
+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	+∞	-∞	± IND
Xj	+N			+∞	-∞	0	0	IND
	-N			-∞	+∞	0	0	IND
	+0	0	0	IND	IND	0	0	IND
	-0	0	0	IND	IND	0	0	IND
	+∞	+∞	-∞	+∞	-∞	IND	IND	IND
	-∞	-∞	+∞	-∞	+∞	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 left shifted 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 round the results in single-precision computation. These instructions execute 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. The other coefficient is then added into the upper half of the accumulator. The result is a double-length register with the format shown in figure 5-5.

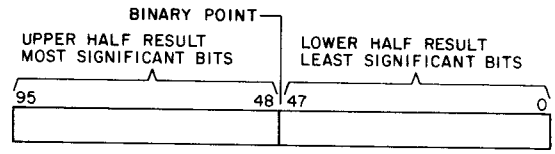


Figure 5-5. Floating-Add Result Format

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 generates from two 48-bit coefficients. The result of a multiply is a double-length register with the format shown in figure 5-6.

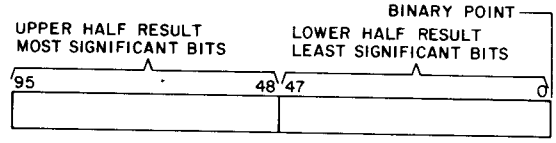


Figure 5-6. Mutiply Result Format

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 effectively moves 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 — MODELS 171 THROUGH 176

Fixed-point addition and subtraction of 60-bit numbers are handled by the long-add instructions (36, 37). Negative numbers are represented in one's-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 one's 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 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.	Normalize X2 in X2 and B0	Normalize X2 (dividend)
5.	Floating quotient of X2 and X0 to Xi	Divide
6.	Unpack X1 to X1 and B7	Unpack quotient
7.	Shift X1 nominally left B7 places	Shift to integer position

The divide requires that both integer (2^{47} maximum) operands be in floating-point format, and the dividend coefficient must be less than two times the divisor coefficient. The normalize X3 instruction ensures this condition.

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 minus (-n) minus 48 equals n minus 48 ≤ 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.

INTEGER ARITHMETIC — MODELS 171 THROUGH 176

Integer divide packs the integers into floating-point format using the pack instruction with a zero-exponent value.

In integer multiplication, a 48-bit product can be formed by using the double-precision multiply instruction. Both operands must have an exponent value of ± 0, and the coefficients cannot both be normalized. The result is sign-extended to 60 bits and sent to an X register.

In integer division, the divisor must be normalized but the dividend need not be normalized. The resulting quotient must be unpacked and the coefficient shifted by the amount of the unpacked exponent using the left shift (22) instruction to obtain the integer quotient.

COMPARE/MOVE ARITHMETIC — MODELS 171 THROUGH 174

The compare/move arithmetic applies to model 171 only if it has the optional compare/move unit.

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 unequal, the characters are referenced in a collation table and the values 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 that is found to be unequal.

PROCESSING DIFFERENCES

Multiply Differences

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. Models 175 and 176 test for the overflow condition by checking for the exponent greater than positive 1777 before correction, if any, is made for a left shift of one. Thus, even through 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 171 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

Models 171 through 174 result:

X0 = 3777 4000 0000 0000 0000

Models 175 and 176 result:

X0 = 3777 0000 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. Models 175 and 176 test for the underflow condition by checking for the exponent 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 171 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

Models 171 through 174 result:

X0 = 0000 4444 4444 4444 4442

Models 175 and 176 result:

X0 = 0000 0000 0000 0000 0000

Floating-Add Differences

The models 171 through 175 floating-add unit may generate a different result from a model 176 floating-add unit when at least one operand has a zero coefficient and the difference between the exponents is greater than or equal to 128 (decimal).

Example: 30012 Floating-Add

X1 = 4277 7777 7777 7777 7777

X2 = 5277 5555 5555 5555 5555

Models 171 through 175 result:

X0 = 4277 7777 7777 7777 7777

Model 176 result:

X0 = 3500 0000 0000 0000 0000

Reversing the operands (30021) gives the same results as indicated previously.

Example: 31012 Floating Difference

X1 = 4277 7777 7777 7777 7777

X2 = 2500 2222 2222 2222 2222

Models 171 through 175 result:

X0 = 4277 7777 7777 7777 7777

Model 176 result:

X0 = 3500 0000 0000 0000 0000

Example: 31012 Floating Difference

X1 = 5277 5555 5555 5555 5555

X2 = 3500 0000 0000 0000 0000

Models 171 through 175 result:

X0 = 4277 7777 7777 7777 7777

Model 176 result:

X0 = 3500 0000 0000 0000 0000

Reversing the operands (31021) on either of the examples for a floating difference gives compatible results on the different models. The result on any model is 3500 0000 0000 0000 0000.

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. Models 175 and 176 test for the underflow condition by checking for the exponent less than negative 1777 before correction, if any, is made for a right shift of one. Thus, even though 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 171 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

Models 171 through 174 result:

X0 = 0000 4000 0000 0000 0000

Models 175 and 176 result:

X0 = 0000 0000 0000 0000 0000

Floating-Divide Condition Differences

If model 176 senses a divide fault, an indefinite condition is indicated only if no overflow or underflow condition exists. If an overflow or underflow condition exists, the divide fault is ignored. If models 171 through 175 sense a divide fault, the fault is identified as an indefinite condition.

Example: 44012

X1 = 3700 0222 0000 0000 0000

X2 = 1600 0022 0000 0000 0000

Models 171 through 175 result:

X0 = 1777 0000 0000 0000 0000 (indefinite condition)

Model 176 result:

X0 = 3777 0000 0000 0000 0000 (overflow condition)

Round-Divide Differences

Models 171 through 175 perform a one-third round. This adds the quantity of one-third to the dividend on the divide. Model 176 performs a one-half round. This adds the quantity of one-half to the divident on the divide. These differences can produce different results for certain operands.

Example: 45012

X1 = 2057 7223 2220 7175 5360

X2 = 1347 4255 6115 0364 7225

Models 171 through 175 result:

X0 = 2430 6557 3505 0613 2700

Model 176 result:

X0 = 2430 6557 3505 0613 2701

Instructions 22 and 23 Differences

When instruction 22 or 23 is used for a right shift, the model 176 checks bits 6 through 11 for a shift greater than or equal to 64 (decimal) and ignores bits 12 through 16. Models 171 through 175 check bits 6 through 10 and ignore bits 11 through 16.

When a negative number is right shifted more than 63 (decimal) places, models 171 through 175 return a positive zero and model 176 returns a negative zero.

ILLEGAL INSTRUCTIONS — MODELS 171 THROUGH 175

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 execute as passes and do not change the content of any register (except as noted in 5 of the following list).

- 011, 012 with no ECS or in parcel 1, 2, 3
- 013 with CEJ/MEJ disabled or in parcel 1, 2, 3
- 014 through 017
- 464 through 467 (model 175), 464 through 467 in parcel 1, 2, 3 (models 172 through 174)
- 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 illegal and aborted. Registers and P values change before the program stops.)

EXIT MODE/ERROR RESPONSE — MODELS 171 THROUGH 175

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 - MODELS 171 THROUGH 175

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	CPU to CMC address or data parity error or CPU to CMC address parity error
53	59	CMC to CPU data parity error or double error

Error condition bits 48, 49, and 50 are detected in the CP, and condition bits 51, 52, and 53 are flags sent to the CP from the CMC. Condition bit 51 indicates a transmission error on the address between the ECS coupler and ECS controller when the ECS flag register operation is being used. Condition bit 52 indicates that a transfer from the CP caused a data or address parity error at CMC or an address parity error at CM. Condition bit 53 indicates a double error on data requested by the CPU in single-error correction double-error detection (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. When an error exit occurs, the content of the P register may not correspond to the address of the instruction that caused the error exit. The P register may have been incremented prior to the execution of the instruction.

Tables 5-7 through 5-9 explain what happend 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.

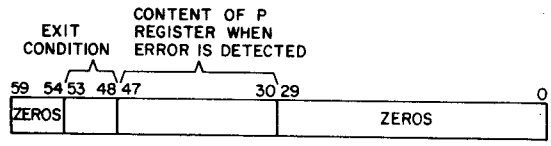


Figure 5-7. Format of Relative Address Zero on Error Exit - Models 171 through 175

TABLE 5-7. ERROR RESPONSE WITH CEJ/MEJ ENABLED, MF SET - MODELS 171 THROUGH 175

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Illegal instruction	1. Execute 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 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 selected X register. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Read all zeros to 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.

TABLE 5-7. ERROR RESPONSE WITH CEJ/MEJ ENABLED, MF SET - MODELS 171 THROUGH 175 (Contd)

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
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 greater than 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 or 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) CMC to CPU data parity error or double error (bit 53)	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.	Continue execution.
CPU to CMC address or data parity error or CPU to CMC address parity error (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.
CPU to CMC address parity error on exchange jump address (bit 52)	1. Write operation is not blocked. 2. Block read operation of first word forces read data to all ones. 3. Stop CP. 4. Store P and exit condition bits at RAC. 5. Clear P.	1. Write operation is not blocked. 2. Block read operation of first word forces read data to all ones. 3. Rest of exchange jump executes normally.
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 - MODELS 171 THROUGH 175

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Illegal instruction	1. Execute 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 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 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 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.
Exit condition bit 48 set on CMU instruction (models 172 through 174 only) 1. C1 or C2 greater than 9 2. C1 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 or 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. Forces 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.

TABLE 5-8. ERROR RESPONSE WITH CEJ/MEJ ENABLED, MF CLEAR - MODELS 171 THROUGH 175 (Contd)

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Infinite condition (bit 49) Indefinite condition (bit 50) ECS flag register parity (bit 51) CMC to CPU data parity error or double error (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.
CPU to CMC address or data parity error or CPU to CMC address parity error (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.
CPU to CMC address parity error on exchange jump address (bit 52)	1. Write operation is not blocked. 2. Block read operation of first word 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. Write operation is not blocked. 2. Block read operation of first word forces read data to all ones. 3. Rest of exchange jump executes normally.
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 - MODELS 171 THROUGH 175

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
Illegal instruction	1. Execute 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 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 selected X register. 2. Stop CP. 3. Store P and exit condition bits at RAC. 4. Clear P.	1. Read all zeros to selected X register. 2. Continue execution. 3. 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 greater than 9 2. C1 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. Forces 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) CMC to CPU data error or double error (bit 53)	1. Stop CP. 2. Store P and exit condition bits at RAC. 3. Clear P.	Continue execution.
CPU to CMC address or data parity error or CPU to CMC address parity error (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 5-9. ERROR RESPONSE WITH CEJ/MEJ DISABLED - MODELS 171 THROUGH 175 (Contd)

Error Condition	Error Response	
	Exit Mode Selected	Exit Mode Not Selected
CPU to CMC address parity error on exchange jump address (bit 52)	1. Write operation is not blocked. 2. Block read operation of first word forces read data to all ones. 3. Stop CP. 4. Store P and exit condition bits at RAC. 5. Clear P.	1. Write operation is not blocked. 2. Block read operation of first word forces read data to all ones. 3. Rest of exchange jump executes normally.
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 PROGRAMMING

CENTRAL MEMORY — MODELS 171 THROUGH 175

All references to CM by the CP for instructions or read/write data are made relative to RAC. The RAC defines the lower limit of the addresses of a program in CM. The upper limit of the program addresses is defined by FLC added to RAC. The field length is a number of 60-bit words established by the operating system prior to program execution. All references to CM for a program must be within the field established for that program.

During an exchange jump, an 18-bit RAC and an 18-bit FLC load into respective registers to define the CM limits of the program that is initiated by the exchange jump.

Figure 5-8 shows the absolute and relative memory addresses, RAC, FLC and P register relationships. For a program to operate within the established limits, the following conditions must exist.

For absolute memory addresses:

$RAC \leq (RAC + P) < (RAC + FLC)$

For relative memory addresses:

$O \leq P < FLC$

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).

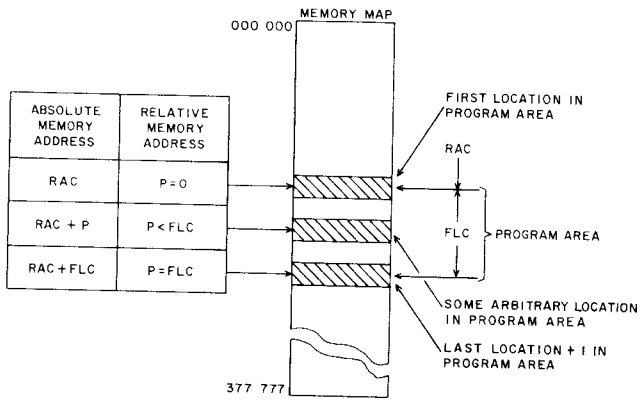


Figure 5-8. Memory Map - Models 171 through 175

CM references beyond the described limits cause error responses listed in tables 5-7, 5-8, and 5-9.

CENTRAL MEMORY — MODEL 176

All references to CM by the CP for instructions or read/write data are made relative to RAS. The RAS defines the lower limit of the addresses of a program in CM. Changes to RAS permit relocation of the program in CM. The upper limit of the program addresses is defined by the FLS added to RAS. The field length is a number of 60-bit words established by the operating system prior to program execution. All references to CM for a program must be within the field established for that program.

During an exchange jump, an 18-bit RAS and an 18-bit FLS load into respective registers to define the CM limits of the program that is initiated by the exchange jump.

Figure 5-9 shows the absolute and relative memory addresses, RAS, FLS and P register relationships. For a program to operate within the established limits, the following conditions must exist.

For absolute memory addresses:

$RAS \leq (RAS + P) < (RAS + FLS)$

For relative memory addresses:

$O \leq P < FLS$

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 ((RAS + FLS) - 1) for an instruction produces a range fault when the RNI occurs to (RAS + FLS). Data, rather than instructions, should always be stored in addresses of absolute locations ((RAS + FLS) - 1) and (RAS + FLS).

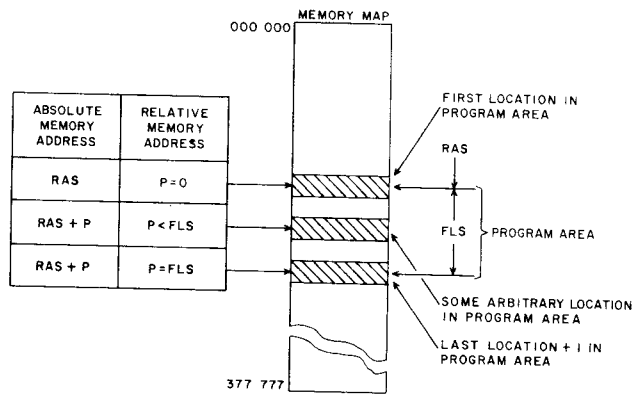


Figure 5-9. Memory Map - Model 176

CM references beyond the described limits set flags in the PSD register, described for the model 176 CP in section 2.

BREAKPOINT — MODELS 171 THROUGH 175

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 stores in bits 72 through 75 and the content of that PP's P register stores in status and control register bits 60 through 71. This status holds until bit 76 clears.

The following breakpoint notes only apply to model 175.

- 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 enters the instruction stack before selecting breakpoint.
- The value of P plus RAC when the CP stops for breakpoint may not correspond with the value of breakpoint address because the CP normally requests two words ahead of P on an RNI.
- The value of P plus RAC when the CP stops for 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.

DATA CHANNEL CONVERTER PROGRAMMING

DATA CHANNEL CONVERTER PROGRAMMING

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) Each DCC is assigned different select and deselect codes, such as 2200 and 2300 and 2400 and 2500, when two or more DCCs share a common data channel.
(2) N equals equipment number 4 through 7, and UUU equals lower nine bits of connect code.
(3) FFF equals lower nine 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 XXXool, where ool 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. Each DCC is assigned different select and deselect codes, such as 2200 and 2300 or 2400 and 2500, when two or more DCCs share a common data channel. A deadstart master clear automatically selects all DCCs in the computer system. The DCC must be the first equipment on a data channel.

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 equals the lower nine bits of the connect code.

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 nine bits of any 12-bit code whose upper three 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 (EOR) signal from the 3000 series equipment or by a channel disconnect from the PP. Initiate conditions are defined by Xoox. A negate BCD conversion line is enabled by the external equipment when bit 0 of code 14Xoox is set.

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 which generates an inactive signal that is 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 and 1 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 sets and remains set until the equipment drops the interrupt signal.

Transmission Parity Error on Data From Channel (1xxXX1xx)

A parity error is 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. DCCs are assigned different select and deselect codes, such as 2200 and 2300 or 2400 and 2500, when two or more 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. The DCC must be the first equipment on a data channel.

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.

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 5-10) connects a 3000 series controller to a DCC.

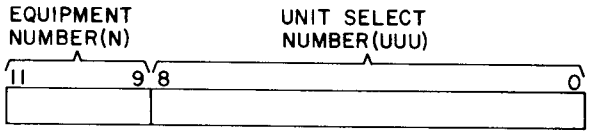


Figure 5-10. DCC Connect Code Format

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 previous 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 a 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 that the addressed equipment successfully connected.

- Reject — Indicates that the addressed equipment could not connect. Reject status bit 0 sets in the DCC.

- No response — The DCC generates an internal reject signal after a 100-microsecond delay. Internal reject status bits 0 and 1 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 a 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 connects. 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 9-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.

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 9-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 equipment accepted the code.
- Reject — Indicates that the equipment did not accept the code. Reject status bit 0 sets in the DCC.
- No response — The DCC generates an internal reject signal after a 100-microsecond delay if neither a reply nor a reject signal is received. Internal reject status bits 0 and 1 set.

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 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 equipment accepted the code.
- Reject — Indicates that the equipment did not accept the code. Reject status bit 0 sets in the DCC.
- No response — The DCC generates an internal reject signal after a 100-microsecond delay if neither a reply nor a reject signal is received. Internal reject status bits 0 and 1 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 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 (15XX) or an input EOR (14XX) initiate function code to the DCC to prepare it for an input operation.
- Execute an active channel (174) 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 (70, 71) instruction to read the data from the sending device.

Input function code 1400 terminates the input operation when either the 3000 series peripheral equipment reaches an end-of-record or 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 binary coded decimal (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 Reader 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 normally followed 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 (16XX) function code to the DCC to prepare it for an output operation.
- Execute an activate channel (74) 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, it starts tape or card motion).
- Execute an output (72, 73) instruction to send data to the 3000 series equipment.
- Execute a full jump (66) instruction 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.

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 one 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 clears. Therefore, the function does not execute.

Data From PPS to DCC

The PPS transmits a 12-bit data byte (includes functional data on mode II connect or function operation) plus one 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 in the channel status word set.
- 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 12 bits of data plus one 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 in the data channel status word sets.
- 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 12 bits of data plus one 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 PROGRAMMING

DISPLAY STATION PROGRAMMING

KEYBOARD

A PP must transmit a one-word function code (7020, octal) to request data from the keyboard of the display station. The code prepares the display controller for an input operation. The PP then activates the input channel and receives one character from the keyboard. This character enters as the lower six bits of the word. The upper bits clear. There is no status report by the keyboard. Table 5-10 lists the keyboard character codes.

TABLE 5-10. KEYBOARD CHARACTER CODES

Character	Code
No data	00
A	01
B	02
C	03
D	04
E	05
F	06
G	07
H	10
I	11
J	12
K	13
L	14
M	15
N	16
O	17
P	20
Q	21
R	22
S	23
T	24
U	25
V	26
W	27
X	30
Y	31
Z	32
0	33
1	34
2	35
3	36
4	37
5	40
6	41

TABLE 5-10. KEYBOARD CHARACTER CODES (Contd)

Character	Code
7	42
8	43
9	44
+	45
-	46
*	47
/	50
(	51
)	52
Left blank key	53
=	54
Right blank key	55
,	56
.	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 octal address X=6000 and Y=7000, and the upper right corner dot is octal 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 5-11 lists the character codes.

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.

TABLE 5-11. DISPLAY CHARACTER CODES

Character	Code
Space	00
A	01
B	02
C	03
D	04
E	05
F	06
G	07
H	10
I	11
J	12
K	13
L	14
M	15
N	16
O	17
P	20
Q	21
R	22
S	23
T	24
U	25
V	26
W	27
X	30
Y	31
Z	32
0	33
1	34
2	35
3	36
4	37
5	40
6	41
7	42
8	43
9	44
+	45
-	46
*	47
/	50
(	51
)	52
Space	53
=	54
Space	55
,	56
.	57

Codes

A single function word is transmitted to select the presentation, mode, and character size (character mode only). Figure 5-11 illustrates the function word format. The word following the function word specifies the starting coordinates for the display (for either mode). Figure 5-12 illustrates coordinate data word. In character mode, the following words are display character codes. Figure 5-13 illustrates the character word.

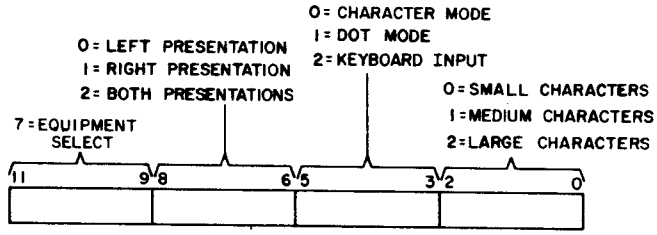
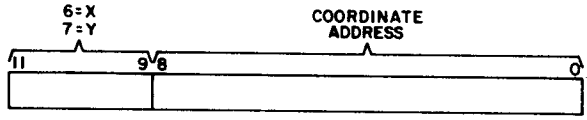


Figure 5-11. Display Station Output Function Code



NOTE:
IN DOT MODE, EACH Y COORDINATE TRANSMITTED FORCES A DOT DISPLAY.

Figure 5-12. Coordinate Data Word

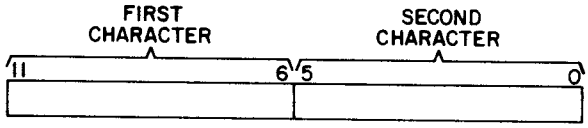


Figure 5-13. Character Data Word

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 5-14) requests an input of one line of data from the display station and displays this data on the CRT as it is being typed.

PROGRAMMING TIMING CONSIDERATION

When performing an output operation at a 2X speed, the computer must wait at the end of the output for a channel empty condition to prevent a loss of coordinates or data. A full jump at the end of the output ensures the channel empty and acceptance by the display controller of the last word of the output before disconnecting from the channel.

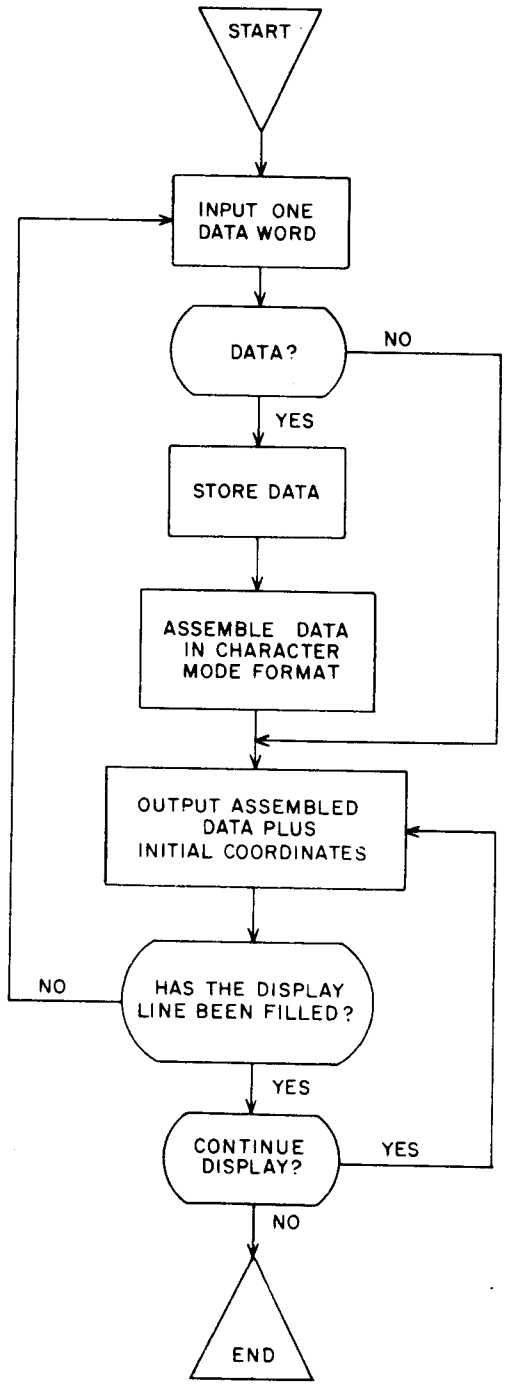


Figure 5-14. Receive and Display Program Flowchart

PERIPHERAL PROCESSOR UNIT PROGRAMMING – MODEL 176

PERIPHERAL PROCESSOR UNIT PROGRAMMING — MODEL 176

RESTRICTIONS ON INSTRUCTION LOOPS

Program loops in the PPU must be four instruction words or longer. There is also a restriction on the number of memory references allowed within the instruction loop. The following criteria may be used for any size instruction loop.

The number of clock periods required to execute each pass through the instruction loop must be equal to or greater than 15N (N is the number of memory references in the instruction loop to any one stack, including the references to obtain instructions in that stack).

In the following example, there are seven references to bank 1, stack 1. The total execution time for the pass is 80 clock periods. Since 15N equals 105 clock periods, this instruction loop violates the restriction.

Address	Instruction	Time (clock periods)	Location
1000	4023	25	0023=1023
1001	4133	25	0033=1033
1002	4243	25	0043=1043
1003	0474	5	

PROGRAMMING CONSIDERATIONS

1. PPU memory parity errors can be caused by deadstarting a PPU while it is executing a program. To eliminate the sensing of such parity errors (false parity errors), the PPU deadstart and loading program should completely load the PPU memory by padding it with zero words and then send the PPU clear parity signal, or cause the PPU to sweep the remainder of the PPU memory and then allow the clear PPU parity error (bit 83 in the status and control register) to be set and clear.

2. If a data sample occurs at the same time that data is changing on the line during a block input, a parity error is likely to occur in the PPU memory.

3. If a MUX output channel resets at the same time that a PPU does a read from the MUX, a parity error is likely to occur in the PPU memory.

4. Deadstarting a PPU while it is executing a program normally destroys one or two words of the program because the deadstart signal clears the X register when the PPU is in the process of a memory read/write cycle. A zero word replaces the word read from memory.

5. The deadstart signal should be applied for a minimum of 32 clock periods when performing a deadstart or dead dump operation on a running PPU. Since the deadstart signal does not clear the shift count register, application of the signal for at least 32 clock periods allows for maximum number of shifts and ensures that shift operations are completed prior to the end of the deadstart or dead dump operation. If the deadstart signal drops before the shift count register clears itself, the content of the A register shifts and becomes incorrect data.

6. The deadstart signal from the status and control register to the PPU (through the scanner) clears the word flags and record flags of the PPU input/output channel. While the deadstart signal is present, all PPU input channel resume flags are forced to ones.

7. A PPU is prepared for a dead dump by sending the deadstart signal, dropping the deadstart signal, sending the dead dump signal, and dropping the dead dump signal. This order of signals ensures the dumping of all PPM locations, excluding location 7777 (octal).

8. If an input record flag is forced to a logical one during a block input, the block input instruction exits and the PPU processes only the data which was on the input channel at the time of the forced record flag.

9. Input channels with forced input word flags can be responsible for PPU parity errors. This occurs when reading data from an input channel directly into memory (71 instruction). This problem is alleviated if all such channel data first goes to the A register (70 instruction) and then to memory from A.

10. If a status word is entered into the A register while the channel input word flag is in a forced logical one condition, the word may have been in a transitional state at the time of entry. To allow for this possibility, consecutively input the status word twice and then compare the two inputs to ensure that they are the same.

11. When terminating a block input instruction by a record pulse, the input record flag remains set until the next input instruction has input at least one word. The last word received during the block input is duplicated in the last block location plus one.

12. When terminating a block output instruction by an output record pulse, the output record pulse should not be sent until the resume for the last word sent has been received. If this is not done, the receiving device (PPU or equipment) may lose the output record pulse and hang (wait for another output record pulse).

13. When the PPU is not selected through the scanner (scanner channel selected for a different PPU), the output channel 0 of the PPU receives a constant output resume.

14. A PPU program cannot reference memory location 7777 (octal).

15. PPU program loops should not be smaller than four words. Program loops smaller than four words may cause marginal memory operation (bit dropping) because of core overheating.

CONTROL SIGNALS

A PPU requires three control signals for I/O communication. The signals include a word pulse, record pulse, and resume pulse. The word pulse must accompany each new data word being transmitted. This pulse signals the receiver that new data is on the data lines. The record pulse is generated by the transmitting device to indicate the end or beginning of a data transmission. The resume pulse is generated by the receiving device to indicate reception of a word pulse and to signify that data will be accepted.

DATA SIGNALS

A PPU data channel has 12 twisted pairs of conductors which carry the 12 data bits in differential mode. In this mode, one conductor of each pair has true data, and the other conductor of the pair has complement data. The characteristic impedance is between 83.3 and 111 ohms.

The PPU holds data stable in its data output register from one word pulse to the next word pulse. The resume pulse signifies that a new word can be entered into the output register but does not clear the register. Transmitting external equipment must hold data stable on the data lines until 55 nanoseconds after the PPU sends a resume pulse.

SEQUENCE TIMING

Figures 5-15 and 5-16 show the input and output channel timing for the control and data signals.

The maximum transfer rate in and out of the PPU is one 12-bit word each 137.5 nanoseconds.

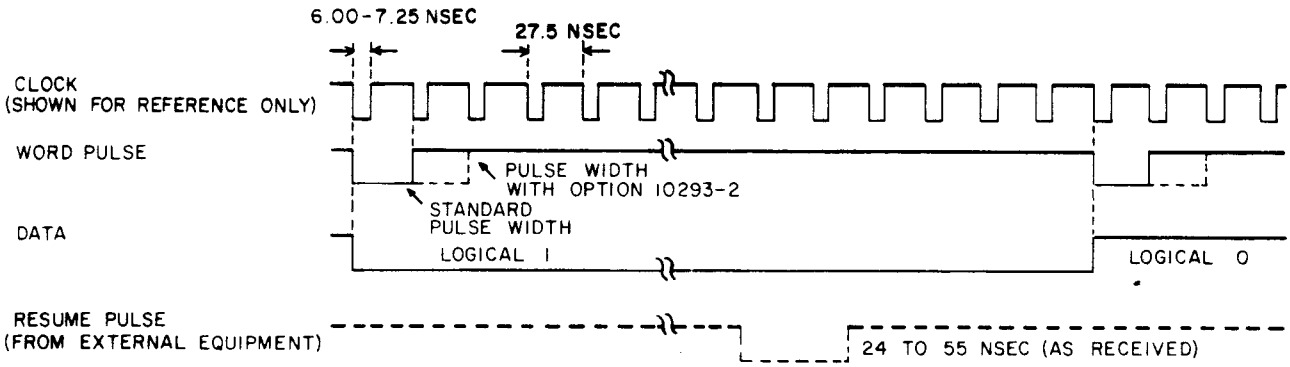


Figure 5-15. Output Channel Timing

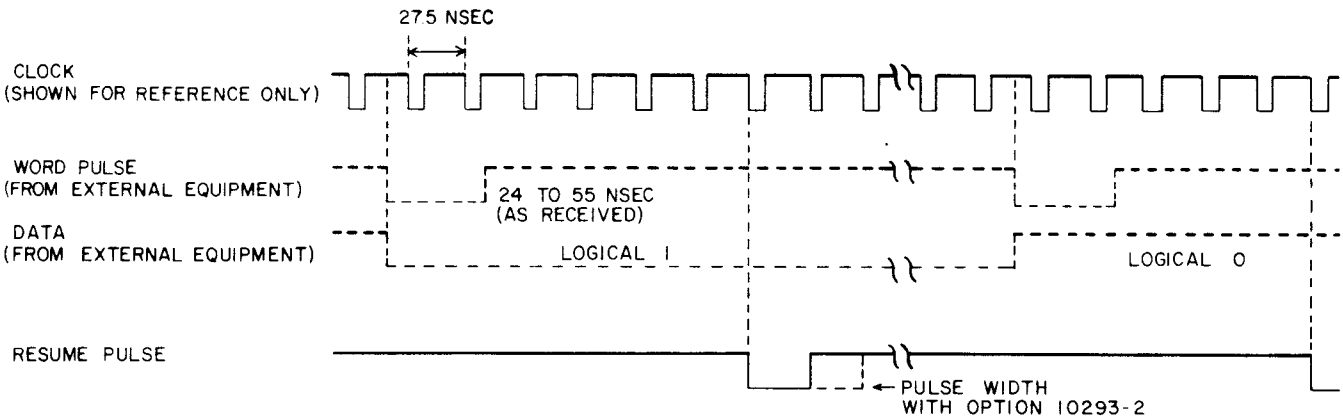


Figure 5-16. Input Channel Timing

PERIPHERAL PROCESSOR PROGRAMMING

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.

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 each 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 no conflicts occur.

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. The PPs are then 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, octal) for the first 12-bit word. The remaining words go to location 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 content of the A register increases 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 at 1X operating speed.

The starting address in CM must enter the A register before the write instruction executes.

For a one-word transfer, the d portion of the write (62) instruction specifies the following.

d is the PPM address (0000 through 0077, octal) of the first 12-bit word. The remaining words are taken from locations 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 content of the A register increases by one with the transfer of each word to provide consecutive CM locations.

CHANNEL DESCRIPTION

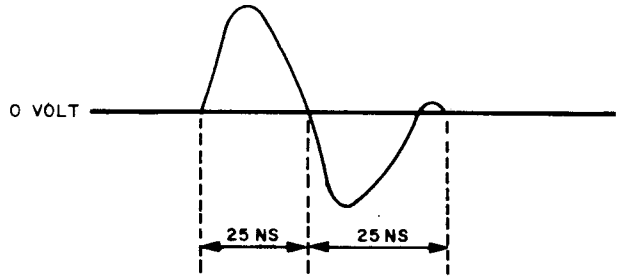
All PPs communicate with each other and with external devices on bidirectional data channels. In a 10-PP system, any PP can access any of 12 data channels (numbered 0 to 13, octal). In a 20-PP system, any PP can access any of 24 data channels (numbered 0 to 13, octal and 20 to 33, octal).

Each data channel has 12 data bits and 1 parity bit, plus several control designators. A channel may connect to one or more external devices, providing the device has data pass-on capability. Only one device can communicate on a channel at one time, but all of the channels can be active at the same time.

Each channel contains a 13-bit bidirectional register (1 parity bit and 12 data bits), a function control signal, and two control flags which are the channel active/inactive flag and the channel register full/empty flag.

Channel Signal Description

Communication between the data channel and the external device is by one-shot, nonrepeat pulses that are synchronized to the PP clock system. A logical one is a pulse (figure 5-17). A logical zero is no signal. Table 5-12 describes the I/O cable line characteristics. Input circuit of the external device must terminate the line in its characteristic impedance and provide storage for the channel data and control signals.



NOTES:
1. MAXIMUM VOLTAGE SWING IS ± 2.7 VOLTS PEAK-TO-PEAK.
2. MINIMUM VOLTAGE SWING IS ± 2.1 VOLTS PEAK-TO-PEAK.
3. VOLTAGE MEASURED AT OUTPUT PIN OF TRANSMITTER, INTO A 75-OHM IMPEDANCE.

Figure 5-17. Channel Output Pulse Characteristics

The data and control lines are grouped into input and output 19-pin coaxial cables for each channel. The input cables carry the external device signals to the PPS and two clocks from the PPS to the external device.

TABLE 5-12. I/O CABLE LINE CHARACTERISTICS

Parameter	Description
Line length, maximum	75-foot (22.9 meters) typical CDC cable
Pulse amplitude at output of transmitter	2.3-volt peak at 32 milliamperes into a 70-73 ohm coaxial cable terminated in its approximate characteristic impedance
Rise time at output of transmitter	2 nanoseconds
Fall time at output of transmitter	2 nanoseconds
Line capacitance	21.5 picofarads/foot (70.5 picofarads/meter) maximum
Line attenuation	0.045 decibel/foot (0.15 decibel/meter) (typical)
Voltage rating	30 volts maximum
Total line length from transmitter to receiver is 75 feet (22.9 meters) including the 5-foot (1.5-meter) cables on the PPS chassis and external equipment.	

The output cables carry PPS signals to the external devices. Table 5-13 lists the signals; the following is a brief description of each.

Data bits 0 through 11 and one data parity bit — Consists of 12 data bits plus one odd parity bit on both input and output cables

Active — Originates from the PPS or the external device to set the channel active flag and reserve a channel for communication

Inactive — Originates from the PPS or the external device to clear the channel full and active flags and terminate communication

Full — Originates from the PPS or the external device to set the register full flag and indicate that a 12-bit data word (plus parity) has been entered in the channel register

Empty — Originates from the PPS or the external device to clear the register full flag and clear the channel register

Function — Originates from the PPS to identify a data transmission as a function code

Clock (10 MHz) — Is a free-running clock transmitted from the PPS to all external devices connected to the data channels. This clock synchronizes the external devices to the PPS. All other signals lag the 10-MHz clock by 25 ± 5 nanoseconds

Clock (1 MHz) — Is a free-running clock transmitted from the PPS to all external devices

Master clear — Is a 1-microsecond train pulse sent at deadstart time to clear all external devices connected to the data channels. This signal is transmitted each 4 milliseconds in the continuous deadstart mode

TABLE 5-13. DATA CHANNEL COAXIAL CABLE LINES

Input Cable	Pin	Color Code	Output Cable
Data bit 0	A	90	Data bit 0
Data bit 1	B	91	Data bit 1
Data bit 2	C	92	Data bit 2
Data bit 3	D	93	Data bit 3
Data bit 4	E	94	Data bit 4
Data bit 5	F	95	Data bit 5
Data bit 6	H	96	Data bit 6
Data bit 7	J	97	Data bit 7
Data bit 8	K	98	Data bit 8
Data bit 9	L	99	Data bit 9
Data bit 10	M	900	Data bit 10
Data bit 11	N	901	Data bit 11
Active	P	902	Active
Inactive	R	903	Inactive
Full	S	904	Full
Empty	T	905	Empty
Clock (10 MHz)	U	906	Function
Clock (1 MHz)	V	907	Master clear
Input data parity	W	908	Output data parity

Channel Operation

External Channel Timing

All control and logic signals occur 25 ±5 nanoseconds following the 10-MHz clock on the channel.

All ac signals transmitted on the channel are 25 ±5 nanoseconds in width.

Worst-case timing requirement, between function and inactive measured at the PPS, to maintain a 500-nanosecond cycle time is 310 ±35 nanoseconds.

Responses from the external device may occur in multiples of 100 nanoseconds greater than 310 nanoseconds, provided that the maximum I/O transfer speed is not required.

Frequency Margins

The PP can vary its master clock frequency ±4 percent. Peripheral devices designed to operate on a channel must tolerate this frequency variation.

Channel Active/Inactive Flag

A channel is normally activated by a function (76, 77) instruction or by an activate channel (74) instruction. The channel can also be activated by an external device.

A function instruction selects the mode of operation in the external device. The instruction places a 12-bit function word plus parity in the channel register and makes the channel active and full. The function word and the function signal are sent to the external device. No active or full signals are sent during a function instruction. The external device accepts the function word and sends an inactive signal which drops the channel active and full flag, clearing the channel register.

An activate channel instruction prepares a channel for data transfer and sends an active signal to the external device. Subsequent input or output instructions transfer data. A disconnect channel instruction after a data transfer returns the channel to an inactive state, and an inactive signal is sent to the external device.

Register Full/Empty Flag

A register is full when it contains a function or data word for an external device or contains a word received from the external device. The register is empty when the flag clears. The flag is turned on or off as the register changes state. A channel can only be full when it is active.

On data output, the processor places a word in the channel register (the channel should be active and empty) and sets a full flag. The data word plus parity and a full signal are sent to the external device. The external device accepts the word and sends an empty signal to the channel which clears the full flag, clearing the channel register. The active and empty status of the channel signal the PP to send the next word to the register.

On data input, the external device sends a word and a full signal to the data channel. The word is placed in the register, and the full flag sets. The PP stores the word and clears the full flag, clearing the data register. An empty signal is sent to the external device signaling it to send the next data word.

Channel Transfer Timing (Even/Odd PPs)

All communication between a PP and a channel occurs during the slot time of the PP. One 50-nanosecond time slot repeats each 500 nanoseconds. Two adjacent PPs can communicate over a channel across a 50-nanosecond boundary. As an example, PP-A places a word in the channel output register and PP-B takes that word and empties the channel in the following 50 nanoseconds.

To maintain a 500-nanosecond transfer rate over a channel, responses from an external device must be received at the PP 310 ±35 nanoseconds after the request is transmitted (figure 5-18). Cable delays total 270 nanoseconds, allowing a worst-case response time of 40 nanoseconds at the external device. These timing relationships are based on an even-numbered PP communicating with the external device. The slot times of even-numbered PPs precede the channel transmission time by 50 nanoseconds. Similarly, the window available for receiving precedes the even-numbered PP slot time by 100 nanoseconds.

The slot time for odd-numbered PPs coincides with the channel transmission and receiving times, allowing a response time of 410 ±35 nanoseconds. The 410-nanosecond response time cannot be used by the external device since it has no control over selection of even or odd PPs.

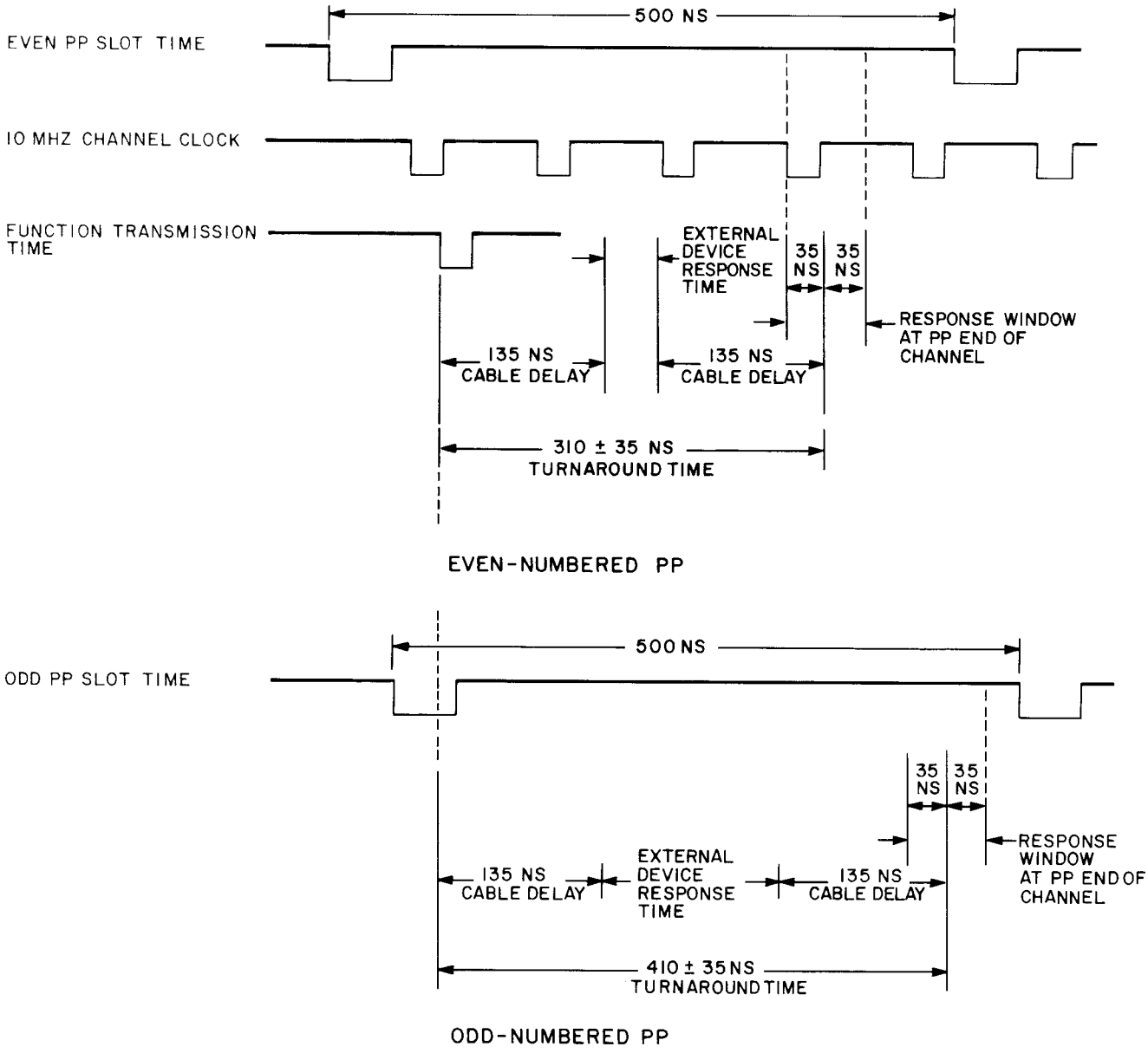


Figure 5-18. Channel Transfer Timing

CHANNEL CONFLICTS IN A BASIC PPS

NOTE

Memory refresh and associated conflicts apply to models A and B only.

2X Operating Speed

In a basic PPS with 10 PPs, channel conflicts may occur at a 2X operating speed only if a memory refresh takes place during a block transfer. During a block transfer over a channel, a block transfer rate of 500 nanoseconds occurs for up to 63 words. A 500-nanosecond refresh cycle then begins for the PPMs. During the refresh cycle, the PPMs act as 10 memories phased 50 nanoseconds apart.

1X Operating Speed

The transfer rates of a basic PPS with 10 PPs operating at a 1X speed are unaffected by memory refresh.

CHANNEL CONFLICTS IN AN EXPANDED SYSTEM

NOTE

Memory refresh and associated conflicts apply to models A and B only.

2X Operating Speed

In an expanded PPS of 14, 17, or 20 PPs (figure 5-19), the possibility of channel conflicts exists under certain conditions. The conflicts cause decreased channel transfer rates and may occur at either the 2X or 1X operating speed.

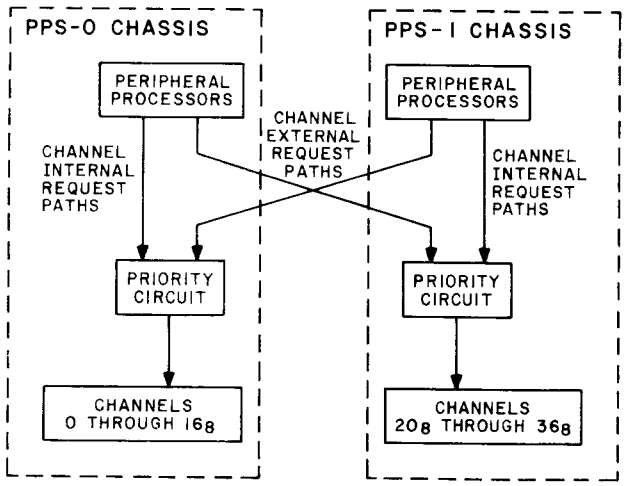


Figure 5-19. PP Channel Request Paths

A PPS operating at 2X speed has two possible channel conflicts, priority and refresh. A priority conflict can result when two PPs (one from each PPS chassis) attempt to access channels in the same PPS chassis at the same time. Table 5-14 further defines the conditions for the conflicts. These conflicts are resolved by a 500-nanosecond toggle circuit in each PPS chassis. The circuit assigns alternate priority to each of the two PPs as long as a channel conflict continues.

TABLE 5-14. CHANNEL CONFLICT CONDITIONS AT 2X OPERATING SPEED

Conflict Conditions	Possible Conflicts
One PP from each PPS chassis accessing channels in PPS-1 (channels 20 through 36 normally or 0 through 16 when reconfigured)	PP0 with PP30 PP1 with PP31 PP2 with PP20 PP3 with PP21 PP4 with PP22 PP5 with PP23 PP6 with PP24 PP7 with PP25 PP10 with PP26 PP11 with PP27
One PP from each PPS chassis accessing channels in PPS-0 (channels 0 through 16 normally or channels 20 through 36 when reconfigured)	PP20 with PP10 PP21 with PP11 PP22 with PP0 PP23 with PP1 PP24 with PP2 PP25 with PP3 PP26 with PP4 PP27 with PP5 PP30 with PP6 PP31 with PP7

The toggling reduces the channel transfer rate from one word each 500 nanoseconds to one word each 1000 nanoseconds. The priority conflicts do not occur when two PPs access channels in their own chassis or when they access channels in the opposite chassis.

Refresh conflicts can occur when a PPM refresh takes place while a PP is executing a 70 through 77 channel instruction in the opposite PPS chassis. A conflict causes the PP to take an additional trip of 500 nanoseconds around the barrel. (Refer to Barrel and Slot in section 2.) The 500-nanosecond delay can extend to 1500 nanoseconds if the refresh and priority conflicts occur back-to-back.

1X Operating Speed

Priority conflicts do not occur in an expanded PPS operating at 1X speed. This is because phasing of the PPs in each PPS chassis permits the PPs to access the channels in the opposite PP's chassis during alternate 500-nanosecond periods.

Refresh conflicts can occur when a PPM refresh takes place while any of PPs 2 through 11 or 22 through 31 are executing a 70 through 77 channel

instruction in the opposite chassis. A conflict causes the PP to take an additional trip of 1000 nanoseconds around the barrel. PPs 0, 1, 20, and 21 are not affected by refresh at 1X speed. None of the PPs are affected by a refresh when they access channels in their own chassis.

Inter-PP Communication of I/O Control

The master/slave I/O drivers utilize changes in the channel control signals to manage transfer of I/O control between the PPs. When utilizing the master/slave I/O drivers for writing, the programmer must consider the operational characteristic of adjacent old/even PPs. A possibility of losing synchronization between the two PPs occurs when the odd-numbered PP senses and reacts to a channel status change and the program does not allow time for the even PP to sense the same change before the channel reverts to its original status. This situation can be prevented if the program does not require synchronization of the two PPs to be accomplished during the same trip around the barrel.

Input/Output Transfers

Data Input Sequence

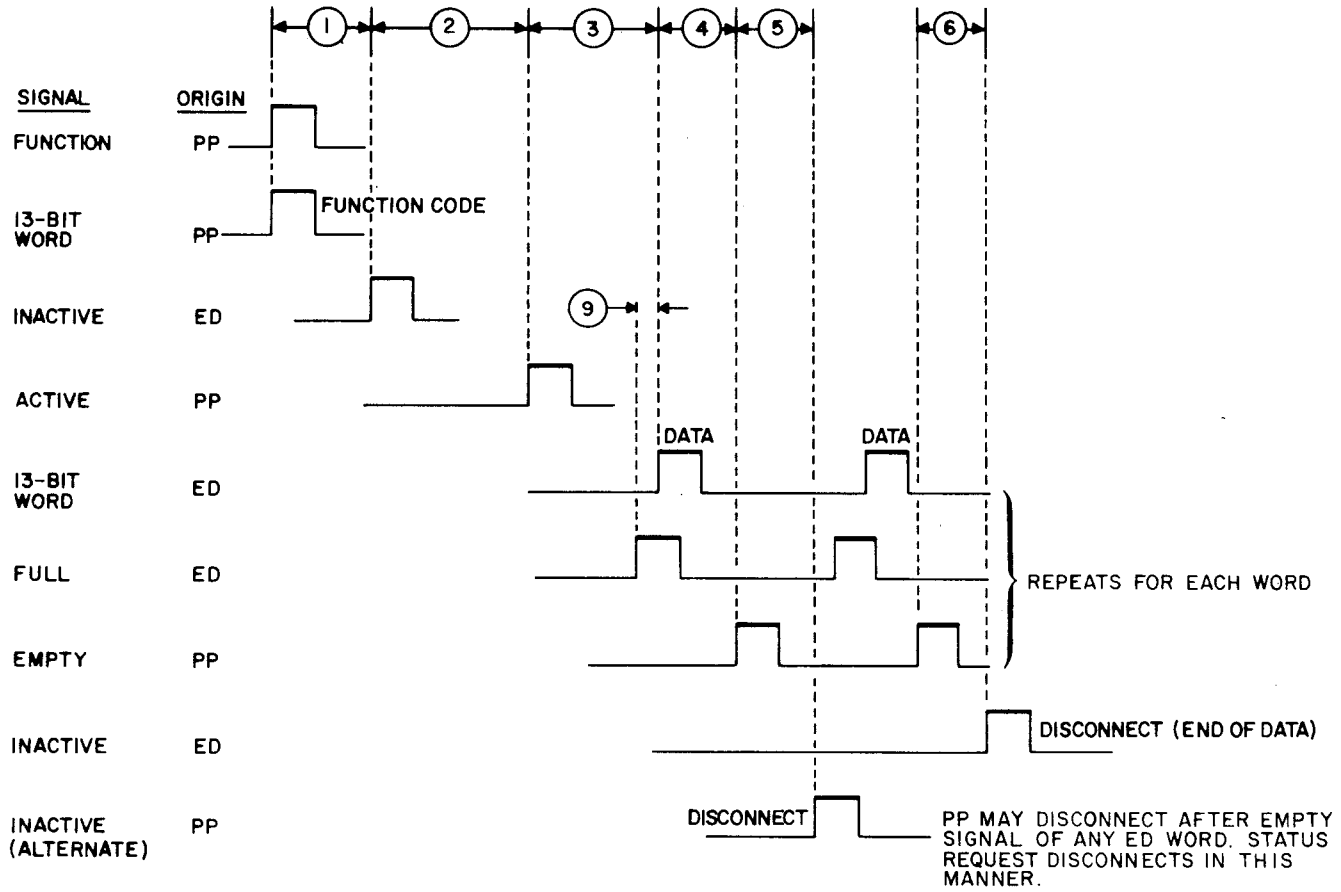
Input equipment sends data (figure 5-20) to the PP by way of the controller as follows:

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 data word plus one 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 (figure 5-21) 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. The function signal causes all controllers to sample the word and identify the word as a function code rather than a data word. 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 one parity bit in the channel register and sets the full flag. At the same time, the PP sends a 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.



NOTES:

(1) TIME IS A FUNCTION OF EXTERNAL DEVICE (ED). PP RECOGNIZES INACTIVE 1 MAJOR CYCLE (OR A MULTIPLE OF MAJOR CYCLES) AFTER FUNCTION. THE PP MUST PREVIOUSLY RECEIVE INACTIVE.

(2) TIME IS A FUNCTION OF PERIPHERAL PROCESSOR (PP). MINIMUM TIME IS 1 MINOR CYCLE. ACTUAL TIME IS A FUNCTION OF THE PP PROGRAM.

(3) TIME IS A FUNCTION OF ED.

(4) TIME IS A FUNCTION OF PP. MINIMUM TIME IS 1 MINOR CYCLE. MAXIMUM TIME IS UP TO 9 MINOR CYCLES TO ALLOW OPERATION WITHIN 1 MAJOR CYCLE.

(5) TIME IS A FUNCTION OF PP. MINIMUM TIME IS 3 MAJOR CYCLES. MAXIMUM TIME IS AN INTEGRAL MULTIPLE OF MAJOR CYCLES.

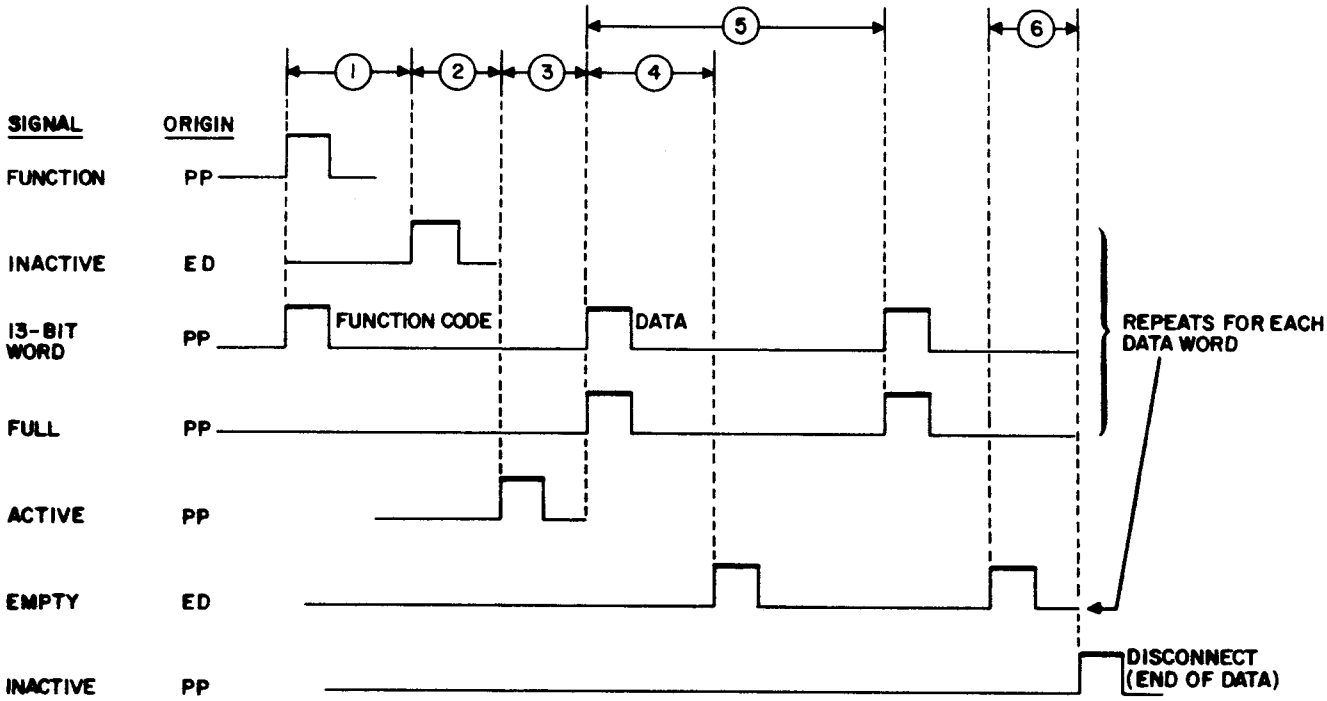
(6) TIME IS A FUNCTION OF ED.

7. MAJOR CYCLE TIME IS 500 NS OR 1 μS (500 NS OR 1 μS MODES OF OPERATION).

8. MINOR CYCLE IS 50 NS OR 100 μS (500 NS OR 1 μS MODES OF OPERATION).

(9) TIME IS A FUNCTION OF ED. FULL SHOULD PROCEED THE DATA BY A MINIMUM OF 5 NS (15 NS MAXIMUM) TO REMOVE THE CLEAR ON THE INPUT DATA RECEIVERS.

Figure 5-20. Data Input Sequence Timing



NOTES:

(1) TIME IS A FUNCTION OF EXTERNAL DEVICE (ED). PP RECOGNIZES INACTIVE 1 MAJOR CYCLE (OR A MULTIPLE OF MAJOR CYCLES) AFTER FUNCTION.

(2) TIME IS A FUNCTION OF PERIPHERAL PROCESSOR (PP). MINIMUM TIME IS 1 MINOR CYCLE. ACTUAL TIME IS A FUNCTION OF THE PP PROGRAM.

(3) TIME IS A FUNCTION OF PP. MINIMUM TIME IS 2 OR 4 MAJOR CYCLES, DEPENDING ON INSTRUCTION. ACTUAL TIME IS A FUNCTION OF THE PP PROGRAM.

(4) TIME IS A FUNCTION OF ED. MINIMUM TIME IS 1 MINOR CYCLE. MAXIMUM TIME IS UP TO 9 MINOR CYCLES TO ALLOW OPERATION WITHIN 1 MAJOR CYCLE.

(5) TIME IS A FUNCTION OF ED. MINIMUM TIME IS 1 MAJOR CYCLE.

(6) TIME IS A FUNCTION OF PP. MINIMUM TIME IS 2 MAJOR CYCLES AFTER EMPTY FROM ED.

7. MAJOR CYCLE TIME IS 500 NS OR 1 μS (500 NS OR 2 μS MODES OF OPERATION).

8. MINOR CYCLE TIME IS 50 NS OR 100 NS (500 NS OR 1 μS MODES OF OPERATION).

Figure 5-21. Data Output Sequence Timing

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 123 to exit from the instruction. The PP then executes the code at P plus 1. The P plus 1 is not necessarily the next instruction.

Force Deadstart

Bit 126 in the status and control register causes the PP selected by bits 120 through 123 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. Software should ensure that the channel is active and empty prior to setting bit 126.

Bit 126 causes the PPS to hang when the selected PP is performing a CM read or write operation at the time of deadstart.

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 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 wired on channel 16 (octal) and located in the PPS-0 chassis.

A second status and control register is present in a 20-PP system. This register is wired to channel 36 (octal) 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 octal function code which tests, clears, and sets the status and control register.

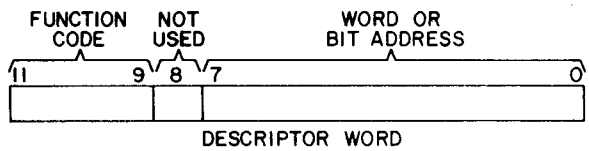


Table 5-15 lists the eight function codes designated by the function code bits.

TABLE 5-15. DESCRIPTOR WORD FUNCTION CODES

Function Code	Function	Function Description
0	Read	The read function reads 1 of 17 words specified by translations 0 through 20 (octal). On the read function, descriptor word bits 0 through 4 designate a 12-bit status and control register word. On all other functions, descriptor word bits 0 through 7 designate a specific status and control register bit.
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 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 the following status and control register bit assignment tables.
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 1 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 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 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 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. Design restrictions also 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).

The control bits of the status and control register have outputs which enable various conditions in the computer. Some 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 always active
FNC 77	Hangs the PP because the channel is always active

STATUS AND CONTROL REGISTER BIT DESCRIPTIONS — MODELS 171 THROUGH 175

STATUS AND CONTROL REGISTER BIT DESCRIPTIONS — MODELS 171 THROUGH 175

Table 5-16 provides a summary of the status and control register bit information for PPS-0 and PPS-1. The table also lists the applicability of the bits to the models.

The following list explains table 5-16.

Column	Information
Word No.	Register word listed in octal (8)
Bit No.	Register bit listed in decimal (10) and octal (8)
Description	Name of bit
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 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.

Column	Information
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 clears at deadstart.
No abbreviation	Read, test, clear, test/clear, set, test/set, and clear all.
Channel 36	X indicates the bit is also used in the abbreviated status and control register of the optional PPS-1.
Display	X indicates that a light-emitting diode displays the bit on a module in PPS-0. When there is an adjacent X in the channel 36 column, the bit is similarly displayed in the optional PPS-1.

TABLE 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	Remarks
0	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 locks bits 40 through 53, 190, and 191
	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
	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	Applies only to models A and B with CM refresh
	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

TABLE 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	Remarks
1	12	14	CP-0 register parity error	X	X	S	TE	X	X	
	13	15	CP-1 register parity error	X		S	TE	X	X	Used only in dual-CP models
	14	16	PP0 parity error	X	X	S	TE	X	X	
	15	17	PP1 parity error	X	X	S	TE	X	X	
	16	20	PP2 parity error	X	X	S	TE	X	X	
	17	21	PP3 parity error	X	X	S	TE	X	X	
	18	22	PP4 parity error	X	X	S	TE	X	X	
	19	23	PP5 parity error	X	X	S	TE	X	X	
	20	24	PP6 parity error	X	X	S	TE	X	X	
	21	25	PP7 parity error	X	X	S	TE	X	X	
	22	26	PP8 parity error	X	X	S	TE	X	X	
	23	27	PP9 parity error	X	X	S	TE	X	X	
2	24	30	Channel 0 parity error	X	X	S	TE	X	X	For channel 36, channel numbers 20 through 33 (octal) apply
	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	
	29	35	Channel 5 parity error	X	X	S	TE	X	X	
	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 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	Remarks
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 15 for M171 through M174, bit 16 for M175	X	X	S	R		X	
	52	64	Syndrome address bit 16 for M171 through M174, bit 17 for M175	X	X	S	R		X	
	53	65	Syndrome address bit 17 for M171 through M174, bit 3 for M175	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	

TABLE 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	Remarks
5	60	74	PPS P register bit 0	X	X	S	R	X	X	If bit 83 clears, bits 60 through 71 display P of the PP selected by bits 120 through 123, and bits 72 through 75 display selected PP.
	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	If bit 83 sets, the content of P register is latched and retained on every CM breakpoint bit.
	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	If bit 76 sets when bit 83 sets, bits 60 through 75 hold until bit 76 clears. Refer to text for more detailed information.
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	
	77	115	CMC breakpoint match	X	X	S		X	X	Loads and locks bits 56 through 59
	78	116	Clear CM busy							
	79	117	Set C5 full							
	80	120	Force zero parity on channels	X	X	C	D	X		
	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

TABLE 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	Remarks
7	84	124	All PPs 500-nano-second major cycle	X	X	C	C		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	Stop on double SECDED error, on a PP read error, or on a read pyramid parity error	X	X	C	D	X	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 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	Breakpoing address bit 11	X	X	C				

TABLE 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	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	Double error on a PP read	X	X	S	D	X	X	
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 equals 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 clears.
	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

TABLE 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	Remarks
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				Applies only to models A and B with CM refresh
	135	207	Refresh margin fast	X	X	C				
	136	210	ECS transfer error code 0	X	X	S	R			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 address/data parity error	X	X	S	R		X	Loaded and locked by bit 5. Clear equals 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			For bit 143, clear equals slow
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 equals lo
	151	227	RVM all/one		X	S			X	Clear equals 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 equals lo
	155	233	Select all/one RVM		X	C				Clear equals one (refer to text)

TABLE 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	Remarks
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 5-16. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODELS 171 THROUGH 175 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	Models 171/172/ 173/174	Model 175	S/C	PRGM FCTN	Channel 36	Display	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 dual-CP models
	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 dual-CP models
	188	274	Software flag 0	X	X	C		X		Diagnostic aids
	189	275	Software flag 1	X	X	C		X		
	190	276	Syndrome address bit 3, models 171 through 174 (bit 4, model 175)	X	X	S	R		X	Loaded and locked by bit 3
	191	277	Syndrome address bit 4, models 171 through 174 (bit 5, model 175)	X	X	S	R		X	
20	192	300	CP-0 stopped	X	X	S	R		X	
	193	301	CP-1 stopped	X		S	R		X	Used only in dual-CP models
	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 dual-CP models
	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							

In the following status and control register bit descriptions, the bit names are preceded by their decimal/octal bit numbers. The decimal numbers are only for reference. The octal numbers are for use in programming, setting, clearing, and testing the bits. The bit functions, status or control, follow each bit name.

BIT 0/0 READ PYRAMID PARITY ERROR — STATUS

This bit indicates a parity error on data transmitted from CMC to PPS. The bit also indicates a CM parity error on data to the PPS during the parity mode operation. The bit may set at the same time a SECDED double error is indicated (bit 3).

BIT $1/1_8$ CSU-0 ADDRESS PARITY ERROR — STATUS

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

BIT $2/2_8$ CSU-1 ADDRESS PARITY ERROR — STATUS

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

BIT $3/3_8$ SECDED ERROR — STATUS

This bit indicates that a single-error correction or a double-error detection has occurred. The bit also locks bits 40 through 53, 190, and 191. These bits are unlocked but not 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 $4/4_8$ NOT USED

BIT $5/5_8$ CMC PARITY ERROR — 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 clears. Bit 5 must be reset by software to detect further CMC parity errors.

BIT $6/6_8$ PARITY ERROR ON DATA RECEIVED FROM EXTERNAL CHANNEL — STATUS

This bit, in an expanded PPS, indicates that a PP in one PPS chassis detected an inter-PPS parity error while executing an input instruction from a channel in the other PPS chassis. The bit sets in the status and control register in the chassis that contains the PP which detected the parity error. Two-way channel communications permit inter-PPS parity errors to be indicated to the status and control register on channels 16 and 36.

BITS $7/7_8$ PARITY ERROR ON DATA TRANSMITTED FROM EXTERNAL PP — STATUS

This bit, in an expanded PPS, indicates that a PP in one PPS chassis detected an inter-PPS parity error on data transmitted from a PP that was executing an output instruction in the other PPS chassis. The bit sets in the status and control register in the chassis that contains the channel receiving the data. Two-way communications permit inter-PPS parity errors to be indicated to the status and control registers on channels 16 and 36.

BITS $8/10_8$ AND $9/11_8$ CSU-0 FAULT AND CSU-1 FAULT — 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.

These bits apply only to models A and B.

Bit $10/12_8$ ERROR IN SECOND PPS — STATUS

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

BIT $11/13_8$ ECS ERROR — 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.

BIT $12/14_8$ CP-0 P REGISTER PARITY ERROR — STATUS

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

BIT $13/15_8$ CP-1 P REGISTER PARITY ERROR — STATUS

This bit applies only to models with two CPs and is unused in models with one CP. The bit indicates that the PPS detected a parity error on a read of the P register for CP-1.

BITS $14/16_8$ THROUGH $23/27_8$ PP0 THROUGH PP9 PARITY ERROR — STATUS

These bits indicate the occurrence of a PP error condition. The bits are used to prevent a PP from executing instructions following the detection of the error condition. On a one-to-one basis, the bits indicate the status of each PP. The bits indicate the logical PP numbers and are not affected by a reconfiguration of the PPMs that results from resetting the PPS-0/PPS-1 and PP MEMORY SELECT switches on the deadstart panel.

The error conditions which can stop the PPs and their associated status and control register bits are:

Error Condition	Bit 0	Bit 119
PPM parity error	0	0
Read pyramid parity error on a CM read	1	0
Double SECDED error on a CM read	0	1

The PP associated with the error stops only if the appropriate enable bits are set in the status and control register. If the enable bits are not set, the error condition is reported but the PP is not stopped.

BITS $24/30_8$ THROUGH $35/43_8$ CHANNEL 0 THROUGH 13 PARITY ERROR — 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 listed in table 5-16. The checking of these bits may be disabled on any or all of the I/O channels with the PARITY switches on the I/O connector panel.

BIT $36/44_8$ MAINS POWER FAILURE — STATUS

This bit indicates that the primary power mains feeding the computer system are deenergized and have remained so for more than one-half cycle (8.3 milliseconds for 60-Hz power and 10.0 milliseconds for 50-Hz power) of the mains frequency. If power returns within one cycle of the mains frequency, the line feeding the bit automatically goes false.

BIT $37/45_8$ SHUTDOWN IMMINENT — STATUS

This bit indicates one of the following conditions.

- The 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.
- An environmental condition (including dew-point warning and chassis temperature) is abnormal and approaching an emergency power shutdown.
- An environmental condition is changing at an abnormally high rate.
- An environmental condition is about to execute a controlled power shutdown.
- 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.

- A power mains failure has occurred for longer than 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.
- A controlled shutdown limit has been reached. The limit sensor has disconnected the primary power mains from the system secondary power supply, and the kernal 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.

	Condition	Explanation
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 capability exists, 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 $38/46_8$ AND $39/47_8$ NOT USED

BITS $40/50_8$ THROUGH $47/57_8$ SYNDROME BITS 0 THROUGH 7 — STATUS

These and bits 48 through 53, 190, and 191 are provided by CMC upon detection of a SECDED error. Bits 40 through 47 provide the information needed to isolate a single-error failure to a particular memory module. Setting bit 3 locks bits 40 through 47. Clearing bit 3 unlocks bits 40 through 47 but does not clear them. Software functions cannot clear or set the read-only syndrome bits.

BITS $48/60_8$ THROUGH $53/65_8$ SYNDROME BITS 0 THROUGH 5 — STATUS

These and bits 40 through 47, 190, and 191 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. Bits 48 through 53 are read-only bits that cannot be cleared or set by software functions.

BITS $54/66_8$ AND $55/67_8$ PARITY ERROR PORT CODE BITS 0, 1 — 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 clears.

Bit 55	Bit 54	Port
0	0	CP-1 (models with two CPs)
0	1	CP-0
1	0	PPS-1
1	1	PPS-0

BITS $56/70_8$ AND $57/71_8$ BREAKPOINT PORT CODE BITS 0, 1 — 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 clears.

Bit 57	Bit 56	Port
0	0	CP-1 (models with two CPs)
0	1	CP-0
1	0	PPS-1
1	1	PPS-0

BITS $58/72_8$ AND $59/73_8$ BREAKPOINT FUNCTION CODE BITS 0, 1 — 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 clears.

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

BITS $60/74_8$ THROUGH $71/107_8$ PPS P REGISTER BITS 0 THROUGH 11 — STATUS

These bits indicate the content of the P register. The content can be the program address or data buffer address 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 on the PPS module located at J40 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 clears.

BITS $72/110_8$ THROUGH $75/113_8$ PP CODE BITS 0 THROUGH 3 — 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 clears. Bit 83 is associated with bits 72 through 75.

BIT $76/114_8$ PPS BREAKPOINT BIT — STATUS

This bit, with bit 83 set, indicates that the breakpoint address was referenced by PPS. The content of the P register of the referencing PP is locked into bit positions 60 through 71. The referencing PP code is also 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.

BIT $77/115_8$ CMC BREAKPOINT MATCH — 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 clears. Bit 77 must be reset by software to detect further breakpoint conditions.

BIT $78/116_8$ CLEAR CENTRAL MEMORY BUSY — 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.

BIT $79/117_8$ SET C5 FULL — 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.

BIT $80/120_8$ FORCE ZERO PARITY ON CHANNELS — CONTROL

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

BIT $81/121_8$ FORCE ZERO PARITY ON PP MEMORIES — CONTROL

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

BIT $82/122_8$ NOT USED

BIT $83/123_8$ PPS BREAKPOINT MODE SELECT — 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.

BIT $84/124_8$ ALL PPs 500-NANOSECOND MAJOR CYCLE — 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.

BIT 85/125$_8$ INHIBIT PPS REQUEST TO CMC — CONTROL

This bit prevents any PP from making a read/write/exchange request. This bit should be used with the CP master clear bit 127 to ensure that the master clear does not hang any PP that is accessing CM. A deadstart clears the bit.

BITS 86/126$_8$ THROUGH 93/135$_8$ NOT USED

BIT 94/136$_8$ STOP ON CM READ ERROR — CONTROL

This bit, when set, enables the stop on a CM read error which may be either a double error or a pyramid parity error. The PP associated with the error stops after disassembling the word received from CM. The data with the error is written into the PPM. In case of a block read, the instruction terminates. Bits 14 through 23 are used to identify the PP with the error condition(s). Bit 94 provides control only in its respective status and control register which is for PPS-0 or PPS-1.

BIT 95/137$_8$ STOP ON PPM PARITY ERROR — CONTROL

This bit, when set, enables the stop on PPM parity error network.

When a parity error is detected, any instruction except a CM read or write, exchange jump, or channel executes. Following the completion of the instruction, the PP stops.

When a parity error is detected on a channel instruction, the channel select control disables, preventing the instruction from performing any channel operation. The instruction may or may not exit.

When a parity error is detected in a 20-PP system and a PP in one chassis is making a request for a channel in the other chassis, the request to the other chassis is blocked. The PP with the parity error hangs in the instruction it is trying to execute.

When a parity error is detected on a CM write or exchange jump instruction, requests to the control of the CM are blocked. A single-word write and exchange exits, and the block write instruction terminates. In all cases, the PP stops prior to writing incorrect data in CM or executing an exchange jump. The instruction is allowed to exit, preventing write pyramid hang-ups.

When a parity error is detected on a CM read instruction, the request is sent and the PP writes the data into PPM. In the case of a block read, the instruction terminates and the PP always stops.

BITS 96/140$_8$ THROUGH 113/161$_8$ BREAKPOINT ADDRESS BITS 0 THROUGH 17 — 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.

BITS 114/162$_8$ THROUGH 117/165$_8$ BREAKPOINT CONDITION CODE BITS 18 THROUGH 21 — 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

BIT 118/166$_8$ INHIBIT SINGLE-ERROR REPORT — 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 119/167$_8$ DOUBLE ERROR ON A PP READ — STATUS

This bit indicates a double SECDED error on a PP read within the PP chassis. Bit 119 functions in conjunction with bits 14 through 23, 94, and 95.

BITS 120/170$_8$ THROUGH 123/173$_8$ PP SELECT CODE BITS 0 THROUGH 3 — 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.

BIT 124/174$_8$ PP SELECT AUTO/MANUAL MODE — CONTROL

This bit selects the mode of PP 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.

BIT 125/175$_8$ FORCE EXIT ON SELECTED PP — CONTROL

This bit clears a selected hung PP (selected by bits 120 through 124) by forcing an instruction exit except in the manual mode. 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.

BIT 126/176$_8$ FORCE PP DEADSTART ON SELECTED PP — 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 clears. A system deadstart clears bit 126.

Bit 126 causes the PPS to hang when the selected PP is performing a CM read or write operation at the time of deadstart.

BIT 127/177$_8$ CSU, CMC, CP MASTER CLEAR — CONTROL

This bit, when set, sends a master clear to the CSU, CMC, and CP chassis (two CP chassis for some models). The bit is ORed 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 bit must be cleared immediately (in less than 6 microseconds) by the program that sets it to prevent the possibility of a memory data error. In special cases on model 175, immediate clearing of the bit also prevents the possibility of a memory fault. To prevent the data errors and memory faults, the PPU programs must minimize the width of the master clear pulse and its occurrence (not more than once each 128 microseconds) and bit 127 should not be used with refresh timing margins in models A and B.

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 bit 127.

BIT 128/200$_8$ FORCE ZERO SECDED CODE AND PARITY CMC TO CM — 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.

BIT 129/201$_8$ FORCE ZERO ADDRESS PARITY CMC TO CM — CONTROL

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

BIT 130/202$_8$ DISABLE ADDRESS PARITY ERROR — 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 131/203$_8$ NOT USED

BITS 132/204$_8$ AND 133/205$_8$ FORCE ZERO PARITY CODE 0 AND CODE 1 — 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	Word count or address from CP to ECS coupler
1	0	Address from ECS coupler to ECS controller
1	1	Data from ECS to CMC

BIT 134/206$_8$ REFRESH MARGIN SLOW — CONTROL

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

This bit applies only to models A and B.

BIT 135/207$_8$ REFRESH MARGIN FAST — CONTROL

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

This bit applies only to models A and B.

BIT 136/210$_8$ THROUGH 138/212$_8$ ECS TRANSFER ERROR CODES THROUGH 2 — 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	CP to CPU address parity error (models 171 through 174)
0	0	1	CP to ECS coupler parity error
0	1	0	CMC double error
0	1	1	CMC to CM address parity error
1	0	0	CMC data 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)

BIT 139/213$_8$ CMC ADDRESS/DATA PARITY ERROR — STATUS

This bit indicates an address parity error in CMC. The bit is used with bits 5, 54, and 55. If the bit clears and bit 5 sets, 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 clears.

BIT 140/214$_8$ NOT USED

BITS 141/215$_8$ THROUGH 143/217$_8$ CLOCK FREQUENCY MAGNITUDE 0, 1, AND SLOW/FAST — CONTROL

These bits are used 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 following 3-bit code translations are for programming use. The codes result in the margin conditions listed after the codes. For example, code 000 results in the margin condition normal, code 001 results in condition slow 1, and so on.

Bit 143	Bit 142	Bit 141
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Margin Condition	40-MHz Clock	20-MHz Clock	10-MHz Clock
Normal	40.000	20.000	10.000
Slow 1	39.375	19.688	9.844
Slow 2	38.750	19.375	9.688
Slow 3	38.125	19.063	9.531
Normal	40.000	20.000	10.000
Fast 1	40.625	20.313	10.156
Fast 2	41.250	20.625	10.313
Fast 3	41.875	20.938	10.469

BITS $144/220_8$ THROUGH $149/225_8$ REFERENCE VOLTAGE MARGIN ADDRESS BIT STATUS — STATUS

These bits apply only to model 175 and are unused in models 171 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.

BITS $150/226_8$ AND $151/227_8$ REFERENCE VOLTAGE MARGIN HI/LO AND ALL/ONE — STATUS

These bits apply only to model 175 and are unused in models 171 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.

BITS $152/230_8$ AND $153/231_8$ CLOCK PULSE WIDTH NARROW AND WIDE — CONTROL

These bits apply only to model 175 and are unused in models 171 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	Wide

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.

BIT $154/232_8$ SELECT HI/LO REFERENCE VOLTAGE MARGINS — CONTROL

This bit applies only to model 175 and is unused in models 171 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.

BIT $155/233_8$ SELECT ALL/ONE REFERENCE VOLTAGE MARGINS — CONTROL

This bit applies only to model 175 and is unused in models 171 through 174. When this bit sets and bits 163 through 173 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.

BIT $156/234_8$ THROUGH $167/247_8$ REFERENCE VOLTAGE MARGINS QUADRANT SELECT 0 THROUGH 11 — CONTROL

These bits apply only to model 175 and are unused in models 171 through 174. The bits determine, on a one-to-one basis, which quadrant(s) (figure 5-22) of a CP chassis receives 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.

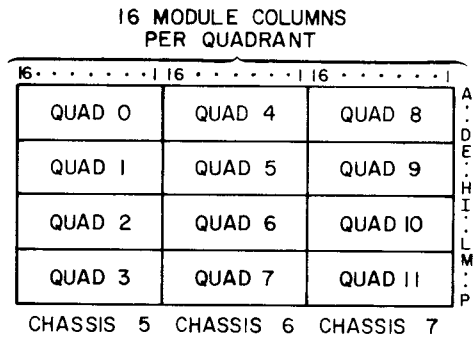


Figure 5-22. CP Chassis Quadrants (Viewed from Module Side) - Model 175

BITS $168/250_8$ THROUGH $173/255_8$ REFERENCE VOLTAGE MARGINS MODULE ADDRESS BITS 0 THROUGH 5 — CONTROL

These bits apply only to model 175 and are unused in models 171 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 one of four module rows. The addresses increase by module location within a row and by rows within a quadrant.

BIT $174/256_8$ PPS TO CMC ZERO ADDRESS PARITY — CONTROL

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

BIT 175/257$_8$ PPS TO CMC ZERO DATA PARITY — CONTROL

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

BITS 176/260$_8$ THROUGH 182/266$_8$ NOT USED

BIT 183/267$_8$ DOUBLE ERROR — STATUS

This bit, when set, indicates that a double error occurred. Software must reset (clear) the bit. When the bit clears and bit 3 sets, 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 sets, indicating a SECDED error.

  Bit 183 clears, 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.

  Bits 190 and 191 indicate the failing chip enable of the failing CM pak.

- A double-bit error occurred.

  Bit 3 sets, indicating a SECDED error.

  Bit 183 sets, 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.

  Bits 190 and 191 indicate the failing chip enable of the failing CM pak.

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

  Bit 3 sets, indicating a SECDED error.

  Bit 183 sets, 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.

  Bits 190 and 191 indicate the failing chip enable of the failing CM pak.

BIT 184/270$_8$ CP-0 TO CMC ZERO ADDRESS PARITY — CONTROL

This bit applies only to models 171 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.

BIT 185/271$_8$ CP-1 TO CMC ZERO ADDRESS PARITY — CONTROL

This bit applies only to models with two CPs and is unused in models with one CP. The bit forces CP-1 to put a zero parity bit on the address sent to CMC.

BIT 186/272$_8$ CP-0 TO CMC ZERO DATA PARITY — CONTROL

This bit applies only to models 171 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.

BIT 187/273$_8$ CP-1 TO CMC ZERO DATA PARITY — CONTROL

This bit applies only to models with two CPs and is unused in models with one CP. The bit forces CP-1 to put a zero parity bit on the data sent to CMC.

BITS 188/274$_8$ AND 189/275$_8$ SOFTWARE FLAG 0 AND FLAG 1 — CONTROL

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

BITS 190/276$_8$ AND 191/277$_8$ SYNDROME ADDRESS BITS 3 AND 4 (MODELS 171 THROUGH 174) AND BITS 4 AND 5 (MODEL 175) — STATUS

These and bits 40 through 53 are provided upon detection of a SECDED error. Bits 190 and 191 indicate which chip enable failed on a memory module. Setting bit 3 locks bits 190 and 191. Clearing bit 3 unlocks bits 190 and 191 but does not clear them. Software functions cannot clear or set the read-only syndrome address bits.

BIT 192/300$_8$ CP-0 STOPPED — STATUS

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

BIT 193/301$_8$ CP-1 STOPPED — STATUS

This bit applies only to models with two CPs and is unused in models with one CP. When set, the bit indicates that the CP-1 has stopped. When the CP resumes operation, the bit clears.

BIT $194/302_8$ ECS IN PROGRESS FLAG — STATUS

This bit indicates ECS transfer is currently in progress. When the transfer completes or terminates, the bit clears.

BIT $195/303_8$ MONITOR FLAG CP-O — STATUS

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

BIT $196/304_8$ MONITOR FLAG CP-1 (BIT 196) — STATUS

This bit applies only to models with two CPs and is unused in models with one CP. The bit indicates the condition of the monitor flag in CP-1.

BIT $197/305_8$ THROUGH $201/311_8$ PPM RECONFIGURATION BITS 0 THROUGH 4 (BITS 197 THROUGH 201) — STATUS

These bits indicate the positions of the PPS-0/PPS-1 and PP MEMORY 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 $202/312_8$ AND $203/313_8$ NOT USED

STATUS AND CONTROL REGISTER BIT DESCRIPTIONS— MODEL 176

STATUS AND CONTROL REGISTER BIT DESCRIPTIONS— MODEL 176

Table 5-17 provides a summary of the status and control register bit information for PPS-0 and PPS-1. The table also lists the applicability of the bits to the models.

The following list explains table 5-17.

Column	Information
Word No.	Register word listed in octal (8)
Bit No.	Register bit listed in decimal (10) and octal (8)
Description	Name of bit
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.

Column		Information
	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 clears at deadstart.
	No abbreviation	Read, test, clear, test/clear, set, test/set, and clear all.
Channel 36		X indicates the bit is also used in the abbreviated status and control register of the optional PPS-1.
Display		X indicates that a light-emitting diode displays the bit on a module in PPS-0. When there is an adjacent X in the channel 36 column, the bit is similarly displayed in the optional PPS-1.

TABLE 5-17. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODEL 176

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PRGM FCTN	Channel 36	Display	Remarks
0	0	0	Not used					For future enhancement
	1	1	Not used					
	2	2	Not used					
	3	3	CM rank II error	S	TE		X	
	4	4	PPU error	S	TE		X	
	5	5	Not used					For future enhancement
	6	6	PE on data received from external PPS channel	S	TE	X	X	
	7	7	PE on data transmitted from external PPS channel	S	TE	X	X	
	8	10	Not used					For future enhancement
	9	11	Not used					
	10	12	Error in second PPS	S	TE		X	Any of status bits 0 through 39 is set in PPS-1
	11	13	LCME rank II error	S	TE		X	

TABLE 5-17. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODEL 176 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PRGM FCTN	Channel 36	Display	Remarks
1	12	14	Not used					For future enhancement
	13	15	Not used					
	14	16	PP0 memory parity error	S	TE	X	X	
	15	17	PP1 memory parity error	S	TE	X	X	
	16	20	PP2 memory parity error	S	TE	X	X	
	17	21	PP3 memory parity error	S	TE	X	X	
	18	22	PP4 memory parity error	S	TE	X	X	
	19	23	PP5 memory parity error	S	TE	X	X	
	20	24	PP6 memory parity error	S	TE	X	X	
	21	25	PP7 memory parity error	S	TE	X	X	
	22	26	PP8 memory parity error	S	TE	X	X	
	23	27	PP9 memory parity error	S	TE	X	X	
2	24	30	Channel 0 parity error	S	TE	X	X	For channel 36, channel numbers 20 through 33 (octal) apply
	25	31	Channel 1 parity error	S	TE	X	X	
	26	32	Channel 2 parity error	S	TE	X	X	
	27	33	Channel 3 parity error	S	TE	X	X	
	28	34	Channel 4 parity error	S	TE	X	X	
	29	35	Channel 5 parity error	S	TE	X	X	
	30	36	Channel 6 parity error	S	TE	X	X	
	31	37	Channel 7 parity error	S	TE	X	X	
	32	40	Channel 10 parity error	S	TE	X	X	
	33	41	Channel 11 parity error	S	TE	X	X	
	34	42	Channel 12 parity error	S	TE	X	X	
	35	43	Channel 13 parity error	S	TE	X	X	
3	36	44	Mains power failure	S	TE		X	Power/environmental abnormal condition
	37	45	Shutdown imminent	S	TE		X	
	38	46	Not used					For future enhancement
	39	47	Not used					
	40	50	CM syndrome bit 0	S	R		X	
	41	51	CM syndrome bit 1	S	R		X	
	42	52	CM syndrome bit 2	S	R		X	
	43	53	CM syndrome bit 3	S	R		X	
	44	54	CM syndrome bit 4	S	R		X	
	45	55	CM syndrome bit 5	S	R		X	
	46	56	CM syndrome bit 6	S	R		X	
	47	57	CM syndrome bit 7	S	R		X	

TABLE 5-17. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODEL 176 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PRGM FCTN	Channel 36	Display	Remarks
4	48	60	CM error address bit 16	S	R		X	
	49	61	CM error address bit 17	S	R		X	
	50	62	CM error address bit 0	S	R		X	
	51	63	CM error address bit 1	S	R		X	
	52	64	CM error address bit 2	S	R		X	
	53	65	CM error address bit 3	S	R		X	
	54	66	Exchange buffer bias bit 0	C			X	
	55	67	Exchange buffer bias bit 1	C			X	
	56	70	Exchange buffer bias bit 2	C			X	
	57	71	Exchange buffer bias bit 3	C			X	
	58	72	Deadstart PPU	C				
	59	73	Dead dump PPU	C				
5	60	74	PPS P register bit 0	S	R	X	X	If bit 124 is clear, bits 60 through 71 display the P register of PP selected by external switches. If bit 124 is set, bits 60 through 71 display the P register of the PP selected by bits 120 through 123.
	61	75	PPS P register bit 1	S	R	X	X	
	62	76	PPS P register bit 2	S	R	X	X	
	63	77	PPS P register bit 3	S	R	X	X	
	64	100	PPS P register bit 4	S	R	X	X	
	65	101	PPS P register bit 5	S	R	X	X	
	66	102	PPS P register bit 6	S	R	X	X	
	67	103	PPS P register bit 7	S	R	X	X	
	68	104	PPS P register bit 8	S	R	X	X	
	69	105	PPS P register bit 9	S	R	X	X	
	70	106	PPS P register bit 10	S	R	X	X	
	71	107	PPS P register bit 11	S	R	X	X	
6	72	110	Scanner select bit 0	C			X	
	73	111	Scanner select bit 1	C			X	
	74	112	Scanner select bit 2	C			X	
	75	113	Scanner select bit 3	C			X	
	76	114	Not used					For future enhancement
	77	115	Deadstart CPU	C				
	78	116	Not used					For future enhancement
	79	117	Not used					
	80	120	Force zero parity on channels	C	D	X		
	81	121	Force zero parity on PPM	C	D	X		
	82	122	Enable scanner interface	C	D		X	
	83	123	Clear PPU parity error	C	D			

TABLE 5-17. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODEL 176 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PRGM FCTN	Channel 36	Display	Remarks
7	84	124	All PPs 500-nanosecond major cycle	C	D		X	
	85	125	Inhibit PPS request to CMC	C	D		X	
	86	126	Block copy exit control	C	D		X	
	87	127	Not used					For future enhancement
	88	130	LCME degrade control bit 0	C	D			
	89	131	LCME degrade control bit 1	C	D			
	90	132	LCME degrade control bit 2	C	D			
	91	133	Reserved for possible LCME expansion	C				
	92	134	Reserved for possible LCME expansion	C				
	93	135	Not used					For future enhancement
	94	136	Stop on CM read error	C	D	X	X	
	95	137	Stop on PPM parity error	C	D	X	X	
10	96	140	LCME error address bit 0	S	R			
	97	141	LCME error address bit 1	S	R			
	98	142	LCME error address bit 2	S	R			
	99	143	LCME error address bit 3	S	R			
	100	144	LCME error address bit 4	S	R			
	101	145	LCME error address bit 5	S	R			
	102	146	LCME error address bit 6	S	R			
	103	147	LCME error address bit 7	S	R			
	104	150	LCME error address bit 8	S	R			
	105	151	LCME error address bit 9	S	R			
	106	152	LCME error address bit 10	S	R			
	107	153	LCME error address bit 11	S	R			

TABLE 5-17. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODEL 176 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PTGM FCTN	Channel 36	Display	Remarks
11	108	154	LCME error address bit 12	S	R			
	109	155	LCME error address bit 13	S	R			
	110	156	LCME error address bit 14	S	R			
	111	157	LCME error address bit 15	S	R			
	112	160	LCME error address bit 16	S	R			
	113	161	LCME error address bit 17	S	R			
	114	162	LCME error address bit 18	S	R			
	115	163	LCME error address bit 19	S	R			
	116	164	LCME error address bit 20	S	R			
	117	165	LCME error address bit 21	S	R			
	118	166	Inhibit CM single-bit error reporting	C				
	119	167	Double error on a PP read	S		X	X	
12	120	170	PP select code bit 0	C	D	X	X	Select 1 of 10 PPs for forced exit, deadstart, or display
	121	171	PP select code bit 1	C	D	X	X	
	122	172	PP select code bit 2	C	D	X	X	
	123	173	PP select code bit 3	C	D	X	X	
	124	174	PP select auto/manual mode	C	D	X	X	Clear equals manual
	125	175	Force exit on selected PP	C	D	X		One-shot operation
	126	176	Force PP deadstart on selected PP	C	D	X		Set forces deadstart. PP remains in deadstart condition until bit clears.
	127	177	CPU clear I/O	C	D			
	128	200	CM configuration status bit 0	S	R			
	129	201	CM configuration status bit 1	S	R			
	130	202	CM configuration status bit 2	S	R			
	131	203	CM configuration status bit 3	S	R			

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PRGM FCTN	Channel 36	Display	Remarks
	132	204	PPU parity error stack 0	S	R		X	
	133	205	PPU parity error stack 1	S	R		X	
	134	206	PPU parity error stack 2	S	R		X	
	135	207	PPU parity error stack 3	S	R		X	
	136	210	PPU program error	S	R		X	
13	137	211	PPU stop on PPU memory parity error or CM error	C				
	138	212	Does not report I/O read errors to CPU	C				
	139	213	Not used					For future enhancement
	140	214	CM test mode	C				
	141	215	Clock frequency margins fast	C				
	142	216	Clock frequency margins slow	C				
	143	217	Clock margin condition	S				
	144	220	LCME syndrome bit 0	S	R		X	
	145	221	LCME syndrome bit 1	S	R		X	
	146	222	LCME syndrome bit 2	S	R		X	
	147	223	LCME syndrome bit 3	S	R		X	
	148	224	LCME syndrome bit 4	S	R		X	
14	149	225	LCME syndrome bit 5	S	R		X	
	150	226	LCME syndrome bit 6	S	R		X	
	151	227	LCME syndrome bit 7	S	R		X	
	152	230	Clock pulse width narrow	C				
	153	231	Clock pulse width wide	C				
	154	232	Select hi/lo RVM	C				Clear equals lo
	155	233	Select all/one RVM	C				Clear equals one (refer to text)
	156	234	CM error address bit 4	S	R		X	
	157	235	CM error address bit 5	S	R		X	
	158	236	CM error address bit 6	S	R		X	
	159	237	CM error address bit 7	S	R		X	
	160	240	CM error address bit 8	S	R		X	
	161	241	CM error address bit 9	S	R		X	
	162	242	CM error address bit 10	S	R		X	
15	163	243	CM error address bit 11	S	R		X	
	164	244	CM error address bit 12	S	R		X	
	165	245	CM error address bit 13	S	R		X	
	166	246	CM error address bit 14	S	R		X	
	167	247	CM error address bit 15	S	R		X	

TABLE 5-17. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODEL 176 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PRGM FCTN	Channel 36	Display	Remarks
16	168	250	CM clear rank II error	C				
	169	251	CM clear rank I error	C				
	170	252	LCME half-zero test	C				
	171	253	LCME parity/SECDED mode	C				
	172	254	LCME maintenance mode	C				
	173	255	LCME test mode	C				
	174	256	CM parity/SECDED mode	C				
	175	257	CM maintenance mode	C				
	176	260	Clear LCME rank II error	C				
	177	261	Clear LCME rank I error	C				
	178	262	Inhibit LCME single-bit error reporting	C				
	179	263	Channels 2, 3 buffer bias bit 0	C			X	
17	180	264	Channels 2, 3 buffer bias bit 1	C			X	
	181	265	Channels 2, 3 buffer bias bit 2	C			X	
	182	266	Channels 2, 3 buffer bias bit 3	C			X	
	183	267	CM double error	S	R		X	
	184	270	Channels 4 through 7 buffer bias bit 0	C			X	
	185	271	Channels 4 through 7 buffer bias bit 1	C			X	
	186	272	Channels 4 through 7 buffer bias bit 2	C			X	
	187	273	Channels 4 through 7 buffer bias bit 3	C			X	
	188	274	Software lock test	C				
	189	275	Software lock clear	C				
	190	276	Reserved for future LCME degrade status	S			X	
	191	277	Reserved for future LCME degrade status	S			X	

TABLE 5-17. STATUS AND CONTROL REGISTER BIT ASSIGNMENTS - MODEL 176 (Contd)

Word No. (8)	Bit No. (10)	Bit No. (8)	Description	S/C	PRGM FCTN	Channel 36	Display	Remarks
	192	300	LCME degrade status bit 0	S	R		X	
	193	301	LCME degrade status bit 1	S	R		X	
	194	302	LCME degrade status bit 2	S	R		X	
	195	303	LCME degrade status bit 3	S	R		X	
	196	304	LCME SECDED double-bit error	S	R		X	
20	197	305	PPM reconfiguration bit 0	S	R			Indicates PPS configuration selected at deadstart panel
	198	306	PPM reconfiguration bit 1	S	R			
	199	307	PPM reconfiguration bit 2	S	R			
	200	310	PPM reconfiguration bit 3	S	R			
	201	311	PPM reconfiguration bit 4	S	R			
	202	312	Not used					For future enhancement
	203	313	Not used					

In the following status and control register bit descriptions, the bit names are preceded by their decimal/octal bit numbers. The decimal numbers are only for reference. The octal numbers are for use in programming, setting, clearing, and testing the bits. The bit functions, status or control, follow each bit name.

BITS 0/0 THROUGH $2/2_8$ NOT USED

BIT $3/3_8$ CM RANK II ERROR — STATUS

This bit indicates a CM parity error (parity mode) or a SECDED error (SECDED mode). Control bit 174 selects the parity mode (set) or SECDED mode (clear). In SECDED mode, bit 183 identifies a SECDED error as a single-bit error when clear or a double-bit error when set. Setting control bit 168 clears the SSM rank II error register. Software must clear status bit 3.

BIT $4/4_8$ PPU ERROR — STATUS

This bit indicates that at least one of the PPUs had a memory parity error or a CM read error or translated an error stop instruction (00 or 77, octal). The failing PPU can be located by selecting each PPU with the scanner select bits 72 through 75 and monitoring PPU error status bits 132 through 136. When the failing PPU is selected, at least one of the status bits 132 through 136 sets to identify the error. After the error is corrected, the set bit must clear.

BIT $5/5_8$ NOT USED

BIT $6/6_8$ PARITY ERROR ON DATA RECEIVED FROM EXTERNAL PPS CHANNEL — STATUS

This bit, in an expanded PPS, indicates that one PPS chassis detected an inter-PPS parity error while executing an input instruction from a channel in the other PPS chassis. The bit sets in the status and control register in the chassis that contains the PP which detected the parity error.

BIT $7/7_8$ PARITY ERROR ON DATA TRANSMITTED FROM EXTERNAL PPS CHANNEL — STATUS

This bit, in an expanded PPS, indicates that one PPS chassis detected an inter-PPS parity error on data transmitted from a PP that was executing an output instruction in the other PPS chassis. The bit sets in the status and control register in the chassis that contains the channel receiving the data.

BITS $8/10_8$ AND $9/11_8$ NOT USED

BIT $10/12_8$ ERROR IN SECOND PPS — STATUS

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

BIT $11/13_8$ LCME RANK II ERROR — STATUS

This bit indicates an LCME parity error when LCME is in parity mode (bit 171 set) or SECDED error when LCME is in SECDED mode (bit 171 clear). In SECDED mode, bit 196 identifies a SECDED error as a single-bit error when clear or a double-bit error when set. Setting control bit 176 clears the LCME rank II register. Software must clear status bit 11.

BITS $12/14_8$ AND $13/5_8$ NOT USED

BITS $14/16_8$ THROUGH $23/27_8$ PP0 THROUGH PP9 MEMORY PARITY ERROR — 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 listed in table 5-17. 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 PP MEMORY SELECT switches on the deadstart panel.

BITS $24/30_8$ THROUGH $35/43_8$ CHANNEL 0 THROUGH 13 PARITY ERROR — 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 listed in table 5-17. The checking of these bits may be disabled on any or all of the I/O channels with the PARITY switches on the I/O connector panel.

BIT $36/44_8$ MAINS POWER FAILURE — STATUS

This bit indicates that the primary power mains feeding the computer system are deenergized 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 within one cycle of the mains frequency, the bit automatically clears. If power does not return within one cycle of the mains frequency, software must clear the bit.

BIT $37/45_8$ SHUTDOWN IMMINENT — STATUS

This bit indicates one of the following conditions.

- The 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.
- An environmental condition (including dew-point warning and chassis temperature) is abnormal and approaching an emergency power shutdown.
- An environmental condition is changing at an abnormally high rate.
- An environmental condition is about to execute a controlled power shutdown.
- 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 within one cycle of the mains frequency, the bit automatically clears. If power and environmental conditions do not return to normal 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.

- A power mains failure has occurred for longer than 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.
- 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.

	Condition	Explanation
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 capability exists, 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 $38/46_8$ AND $39/47_8$ NOT USED

BITS $40/50_8$ THROUGH $47/57_8$ CM SYNDROME BITS 0 THROUGH 7 — STATUS

Bits 40 through 47 represent CM syndrome bits 0 through 7 (SECDED mode) or CM parity error bits 0 through 7 (parity mode). Any set bit corresponds to a CM rank II error. The CM error address bits 0 through 17 (status and control register bits 50 through 53, 156 through 167, 48, and 49) identify the CM word with the error. Control bit 168 clears bits 40 through 47.

BITS $48/60_8$ THROUGH $53/65_8$ AND $156/234_8$ THROUGH $167/247_8$ CM ERROR ADDRESS BITS 0 THROUGH 17 — STATUS

These bits are the address of the CM word that has a parity error or a SECDED rank II error. Control bit 168 clears the bits. The status bits correspond to address bits as follows:

Status Bit	Address Bit
48	16
49	17
50 through 53	0 through 3
156 through 167	4 through 15

BITS $54/66_8$ THROUGH $57/71_8$ EXCHANGE BUFFER BIAS BITS 0 THROUGH 3 — CONTROL

These are exchange address bits 9 through 12 for I/O interrupts.

BIT $58/72_8$ DEADSTART PPU — CONTROL

This bit deadstarts the PPU selected by the scanner select control bits 72 through 75. The selected PPU executes a block input of 4095 words (starting at address 0) on its channel 0.

BIT $59/73_8$ DEAD DUMP PPU — CONTROL

This bit causes the PPU selected by the scanner select control bits 72 through 75 to execute a block output of 4095 words (starting at address 0) on its channel 0. Control bit 58 must first set and then clear.

BITS $60/74_8$ THROUGH $71/107_8$ PPS P REGISTER BITS 0 THROUGH 11 — STATUS

These bits represent the 12-bit P register of the selected PP in the PPS. PP selection is either by four manual switches at location I40 (if control bit 124 is clear) or by selective setting of control bits 120 through 123 under program control (if control bit 124 is set).

BITS $72/110_8$ THROUGH $75/113_8$ SCANNER SELECT BITS 0 THROUGH 3 — CONTROL

These bits select a PPU to communicate with a PP in the PPS. Scanner select 16_8 (bits 0 through 3 set) allows the PPS to use a 12-bit address to select a CP or PPU module for RVM testing (control bits 154 and 155).

BIT $76/114_8$ NOT USED

BIT $77/115_8$ DEADSTART CP — CONTROL

This bit, when set, causes a master clear in the CP. When clear, the bit causes a CP exchange jump to the address determined by the exchange bias bits (control bits 54 through 57).

BITS 78/116$_8$ AND 79/117$_8$ NOT USED

BIT 80/120$_8$ FORCE ZERO PARITY ON CHANNELS — CONTROL

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

BIT 81/121$_8$ FORCE ZERO PARITY ON PP MEMORIES — CONTROL

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

BIT 82/122$_8$ ENABLE SCANNER INTERFACE — CONTROL

This bit, when set, enables the scanner interface circuit in the PPS. When clear, the bit disables the interface circuit so that a PP in the PPS is not able to send or receive the word pulse and resume signals to or from a PPU. Software must set the bit when a PP needs to communicate with a PPU. Software must clear the bit as soon as the communication terminates. A deadstart clears the bit.

BIT 83/123$_8$ CLEAR PPU PARITY ERROR — CONTROL

This bit clears the four-bit parity error register in the PPU selected by the scanner select control bits 72 through 75. Software must clear status bit 4. A deadstart clears the bit.

BIT 84/124$_8$ ALL PPs 500-NANOSECOND MAJOR CYCLE — 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.

BIT 85/125$_8$ INHIBIT PPS REQUEST TO CM — CONTROL

This bit prevents any PP from making a read/write/exchange request. This bit should be used with the CP master clear bit 127 to ensure that the master clear does not hang any PP that is accessing CM. A deadstart clears the bit.

BIT 86/126$_8$ BLOCK COPY EXIT CONTROL — CONTROL

This bit is sent to CP issue control. When clear, a read next instruction (RNI) follows the issue of an 011 or 012 block copy instruction. When set, block copy instructions will exit, and the next instruction in CIW will execute. A deadstart clears the bit.

BIT 87/127$_8$ NOT USED

BITS 88/130$_8$ THROUGH 90/132$_8$ LCME DEGRADE CONTROL BITS 0 THROUGH 2 — CONTROL

A two-million word LCME degrades to one million and to a half-million words. The degradation occurs by manually setting switches in LCME control or by selectively setting control bits 88 through 90 under program control. The degradation method (by switches or program control) is selectable by a manual switch in LCME control.

Bit 88 is the LCME configuration bit. Bits 89 and 90 are the LCME size code bits 0 and 1, respectively. The various valid combinations for bits 88 through 90 and the resulting LCME size and banks selected are listed below.

Size Code Bit 90	Size Code Bit 89	Configuration Bit 88	Memory Size In Millions	Banks Selected
0	0	0	1/2	0, 1
0	0	1	1/2	2, 3
0	1	0	1	0 - 3
0	1	1	1	4 - 7
1	0	0	2	0 - 7
1	0	1	These combinations are not allowed.	
1	1	0		
1	1	1		

A deadstart clears these bits.

BITS 91/133$_8$ AND 92/134$_8$ RESERVED FOR POSSIBLE LCME EXPANSION

BITS 93/135$_8$ NOT USED

BIT 94/136$_8$ STOP ON CM READ ERROR CONTROL

This bit, when set, enables the stop on a CM read which has a double-bit error. The PP associated with the error stops after disassembling the word received from CM. PPM stores the error data. In case of a block read, the instruction terminates. Bits 14 through 23 provide identification of the PP with the error condition(s). Bit 94 provides control only in the respective status and control register for PPS-0 or PPS-1.

BIT 95/137$_8$ STOP ON PPM PARITY ERROR — CONTROL

This bit, when set, enables the stop on PPM parity error network. This feature causes a PP to hang if it references a byte containing a failing bit. The hung PP state occurs when a nonexistent trip count (instruction 50 in trip 7) forces a PP to perform a read mode instruction. The PP stays in the hung

state until software selectively deadstarts that PP (bit 126) or the entire system deadstarts.

When this feature is selected and a PPM parity error is detected, a bit sets in the status and control register to indicate which PPM had the error. This bit remains set until cleared by software.

If parity error occurs, the P register may not indicate the failing address. For example, a parity error may occur on the M portion of a 24-bit jump instruction where the jump is satisfied.

A deadstart clears the bit.

BITS 96/140$_8$ THROUGH 117/165$_8$ LCME ERROR ADDRESS BITS 0 THROUGH 21 — STATUS

These bits represent LCME rank II error address bits. The LCME SECDED rank II error condition (status bit 11) blocks the bits. Setting control bit 176 clears the bit.

BIT 118/166$_8$ INHIBIT CM SINGLE-BIT ERROR REPORTING

This bit, when set while CM is in SECDED mode, prevents single-bit errors from being reported to status bit 3. This bit must be clear when in parity mode (control bit 174 set) or maintenance mode (control bit 175 set).

BIT 119/167$_8$ DOUBLE ERROR ON A PP READ — STATUS

This bit indicates that a double-bit SECDED error occurred on a PP CM read within the PPS chassis. Bit 119 functions in conjunction with bits 14 through 23, 94, and 95.

BITS 120/170$_8$ THROUGH 123/173$_8$ SELECT CODE BITS 0 THROUGH 3 — CONTROL

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

BIT 124/174$_8$ PP SELECT AUTO/MANUAL MODE — CONTROL

This bit selects the mode of PP selection. When set, PP selection is under program control. PP selection is then made by bits 120 through 123. When clear, PP selection is manual, and the PP selection is made by switches on the PPS chassis at location 140. A deadstart clears the bit.

BIT 125/175$_8$ FORCE EXIT ON SELECTED PP — CONTROL

This bit clears a selected hung PP (selected by bits 120 through 124) by forcing an instruction exit except in the manual mode. 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.

BIT 126/176$_8$ FORCE PP DEADSTART ON SELECTED PP — 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 clears. A system deadstart clears bit 126.

Bit 126 causes the PPS to hang when the selected PP is performing a CM read or write operation at the time of deadstart.

BIT 127/177$_8$ CP CLEAR I/O — CONTROL

This bit clears all MUX and PPS requests to CM and clears control flags in bank control.

A deadstart clears the bit.

BITS 128/200$_8$ THROUGH 131/203$_8$ CM CONFIGURATION STATUS BITS 0 THROUGH 3 — STATUS

These bits indicate which 65K quadrants of CM are selected and available for reference. Set bits indicate selections of available quadrants. Bit combinations for quadrant configurations are listed below. Other combinations are undefined.

Bit	Quadrant	65K	131K	196K	262K
128	0	1 0	1 1 0	1 1 1 0	1
129	1	0 1	1 0 1	1 1 0 1	1
130	2	0 0	0 1 1	1 0 1 1	1
131	3	0 0	0 0 0	0 1 1 1	1

BITS 132/204$_8$ THROUGH 135/207$_8$ PPU PARITY ERROR STACK 0 THROUGH 3 — STATUS

These bits indicate which memory stack failed in the PPU selected by the scanner select bits (control bits 72 through 75).

BIT 136/210$_8$ PPU PROGRAM ERROR — STATUS

This bit indicates that the PPU selected by the scanner select bits (control bits 72 through 75) translated an error stop instruction (00 or 77$_8$) or had a CM read error.

BIT 137/211$_8$ PPU STOP ON PPU MEMORY ERROR OR CM ERROR — CONTROL

This bit, when set, enables the PPUs to stop on PPU memory errors or on a CM error signal.

BIT 138/212$_8$ DO NOT REPORT I/O READ ERRORS TO CPU — CONTROL

This bit, when set, prevents the CP from being interrupted by CM errors caused by I/O reads. When clear, the CP is interrupted on all CM read errors.

BIT 139/213$_8$ NOT USED

BIT 140/214$_8$ CM TEST MODE — CONTROL

This bit, when set under program control and with CM in SECDED mode, forces all eight error condition code bits to zero before they are written into CM. An error correction is performed, if necessary.

If CM is in parity mode (control bit 172 set), the bit forces a complement of all eight parity bits before they are written into CM. This feature allows diagnostic programs to force errors which allow SECDED hardware testing.

BITS 141/215$_8$ AND 142/216$_8$ CLOCK FREQUENCY MARGINS FAST AND SLOW — CONTROL

These bits are used in maintenance operations. The PPS has clock frequency margins of plus or minus 4 percent. Frequency margins are selected by manually setting a switch on the clock module (PPS location R29) or by program control setting control bit 141/142 for fast/slow clock. Margin selection under program control has priority over manual selection. Bits 141 and 142 are also sent to the CP and PPU to control margins. If bits 141 and 142 are both set at the same time, frequency margins do not occur.

BIT 143/217$_8$ CLOCK MARGIN CONDITION — STATUS

This bit indicates that clock margins are applied to CP and PPU chassis. The margin condition is because of the setting of control bits 141/142 (fast/slow), control bits 152/153 (narrow/wide), or manual selection through switches.

BITS 144/220$_8$ THROUGH 151/227$_8$ LCME SYNDROME BITS 0 THROUGH 7 — STATUS

These bits represent LCME SECDED syndrome bits 0 through 7. An LCME SECDED double-error locks the bits. Setting control bit 176 clears the bits.

BITS 152/230$_8$ AND 153/231$_8$ CLOCK PULSE WIDTH NARROW AND WIDE — CONTROL

These bits are used in maintenance operations. Bit 152, when set, reduces the clock pulse width on the CP and PPU chassis to 6.25 ±0.25 nanoseconds. Bit 153, when set, widens the CP and PPU clock pulse to 7.25 ±0.25 nanoseconds. When bits 152 and 153 set at the same time, CP and PPU chassis clock pulse width remains at its normal value of 6.75 ±0.25 nanoseconds.

BIT 154/232$_8$ SELECT HI/LO REFERENCE VOLTAGE MARGINS — CONTROL

This bit, when set, selects the high RVM for the CP or PPU module selected by the RVM address bits. When clear, the bit selects the low RVM for the selected module.

BIT 155/233$_8$ SELECT ALL/ONE REFERENCE VOLTAGE MARGINS — CONTROL

This bit enables high or low RVM on all 64 modules in a CP or PPU chassis quadrant. RVM address bits 3 and 7 through 11 select the quadrant. RVM address bits 0 through 2 and 4 through 6 must be set.

BITS 156/234$_8$ THROUGH 167/247$_8$ CM ERROR ADDRESS — STATUS

These bit descriptions are included with descriptions for bits 48 through 53.

BIT 168/250$_8$ CM CLEAR RANK II ERROR — CONTROL

This bit clears a CM rank II error condition. Software must clear the CM rank II error status bit 3.

BIT 169/251$_8$ CM CLEAR RANK I ERROR — CONTROL

This bit clears a CM rank I error condition.

BIT 170/252$_8$ LCME HALF-ZERO TEST — CONTROL

This bit disables the complement control on LCME read to allow maintenance testing of the half-zero complement control circuit in LCME write control.

BIT 171/253$_8$ LCME PARITY/SECDED MODE — CONTROL

This bit disables SECDED and forces LCME to operate in an 8-bit parity mode.

BIT 172/254$_8$ LCME MAINTENANCE MODE — CONTROL

This bit forces reporting of all single-bit errors in LCME while in SECDED mode, the same as double-bit errors.

BIT 173/255$_8$ LCME TEST MODE — CONTROL

This bit, when set under program control and with LCME in SECDED mode, forces all eight error correction code bits to zero before they are written into LCME. An error correction is performed, if

necessary. If LCME is in 8-bit parity mode (control bit 174 set), this bit forces a complement of all eight parity bits before they are written into memory. This feature allows diagnostic programs to force errors to allow SECDED hardware testing.

BIT $174/256_8$ CM PARITY/SECDED MODE — CONTROL

This bit, when set, disables SECDED and forces the CM to operate in an 8-bit parity mode. When clear, the bit enables the CM to operate in SECDED mode.

BIT $175/257_8$ CM MAINTENANCE MODE — CONTROL

This bit forces the reporting of all CM single-bit errors in CM while in SECDED mode, the same as double-bit errors.

BIT $176/260_8$ CLEAR LCME RANK II ERROR — CONTROL

This bit clears the LCME rank II error condition. Software must clear status bit 11.

BIT $177/261_8$ CLEAR LCME RANK I ERROR — CONTROL

This bit clears the LCME SECDED rank I error condition.

BIT $178/262_8$ INHIBIT LCME SINGLE-BIT ERROR REPORTING — CONTROL

This bit, when set while LCME is in SECDED mode, prevents single-bit errors from being reported (status bit 11). The bit must be clear when in parity mode (control bit 174 set) or maintenance mode (control bit 175 set).

BITS $179/263_8$ THROUGH $182/266_8$ CHANNELS 2, 3 BUFFER BIAS BITS 0 THROUGH 3 — CONTROL

These bits go to memory control to form buffer address bits 9 through 12 for MUX channels 2 and 3.

BIT $183/267_8$ CM SECDED DOUBLE-BIT ERROR — STATUS

This bit, when set, indicates that a double-bit error occurred. To enable the bit, CM must be in SECDED mode and status bit 3 must be set. If bit 3 is not set, bit 183 has no useful meaning. When a SECDED error occurs, one of the following conditions describes the error.

- A single-bit error occurred.

  Bit 3 sets, indicating a SECDED error.

  Bit 183 clears, indicating a single-bit error.

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

  Bits 48 through 53 and 156 through 167 indicate the failing CM address.

- A double-bit error occurred.

  Bit 3 sets, indicating a SECDED error.

  Bit 183 sets, indicating a double-bit error.

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

  Bits 48 through 53 and 156 through 167 indicate the failing CM address.

  Error is reported to CP PSD register.

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

  Bit 3 sets, indicating a SECDED error.

  Bit 183 clears, indicating a single-bit error.

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

  Bits 48 through 53 and 156 through 167 indicate the CM address having the single-bit failure.

  A double-bit error is held in CM SECDED rank I error register and indicated to the CP PSD register. The double-bit error is indicated to the status and control register when the single-bit error conditions are cleared by software.

BITS $184/270_8$ THROUGH $187/273_8$ CHANNELS 4 THROUGH 7 BUFFER BIAS BITS 0 THROUGH 3 — CONTROL

These bits go to the I/O MUX to form buffer address bits 9 through 12 for MUX channels 4 through 7.

BITS $188/274_8$ AND $189/275_8$ SOFTWARE LOCK TEST AND CLEAR — CONTROL

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

BITS 190/276$_8$ AND 191/277$_8$ RESERVED FOR FUTURE LCME DEGRADE — STATUS

BITS 192/300$_8$ THROUGH 195/303$_8$ LCME DEGRADE STATUS BITS 0 THROUGH 3 — STATUS

These bits indicate the LCME size and configuration. Bit 192 is the configuration bit, and 193 and 194 are the size code bits 0 and 1, respectively. Bit 195, when set, indicates that the present LCME size and configuration is selected through switches. Bit 195, when clear, indicates that the LCME size and configuration is selected under program control (control bits 88 through 90).

BIT 196/304$_8$ LCME SECDED DOUBLE-BIT ERROR — STATUS

This bit, when set, indicates a double-bit error occurred. To enable the bit, LCME must be in SECDED mode and status bit 11 must be set. If bit 11 is not set, bit 196 has no useful meaning. When a SECDED error occurs, one of the following conditions describes the error.

- A single-bit error occurred.

  Bit 11 sets, indicating a SECDED error.

  Bit 196 clears, indicating a single-bit error.

  Bits 144 through 151 contain the syndrome code (odd number of bits set), indicating the failing bit.

  Bits 96 through 117 indicate the failing LCME address.

- A double-bit error occurred.

  Bit 11 sets, indicating a SECDED error.

  Bit 196 sets, indicating a double-bit error.

  Bits 144 through 151 contain the syndrome code (even number of bits set) that does not indicate the failing bits.

  Bits 96 through 117 indicate the failing LCME address.

  Error is reported to CP PSD register.

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

  Bit 11 sets, indicating a SECDED error.

  Bit 196 clears, indicating a single-bit error.

  Bits 144 through 151 contain the syndrome code (odd number of bits set) indicating the failing bit.

  Bits 96 through 117 indicate the LCME address having the single-bit failure.

  A double-bit error is held in LCME SECDED rank I error register and indicated to the CP PSD register. The double-bit error is indicated to the status and control register when the single-bit error conditions are cleared by software.

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

  Bit 11 sets, indicating a SECDED error.

  Bit 196 sets, indicating a double-bit error.

  Bits 144 through 151 contain the syndrome code (even number of bits set) that does not indicate the failing bits.

  Bits 96 through 117 indicate the LCME address of the first double-bit error.

  The second double-bit error is held in LCME rank I error register and indicated to the CP PSD register. The second double-bit error is transferred to rank II and to the status and control register after rank II is cleared by software.

BITS 197/305$_8$ THROUGH 201/311$_8$ PPM RECONFIGURATION BITS 0 THROUGH 4 — STATUS

These bits indicate the positions of the PPS-0/PPS-1 and PP MEMORY 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 202/312$_8$ AND 203/313$_8$ — NOT USED

GLOSSARY

BPA	Breakpoint address
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
EEA	Exit error address
EM	Exit mode
FLC	Field length for CM
FLE	Field length for ECS
FLL	Field length for LCME
FLS	Field length of program for CM
IAS	Instruction address stack
IFA	Instruction fetch address
I/O	Input/output
IWS	Instruction word stack
LCME	Large core memory extension
MA	Monitor address
MEJ	Monitor exchange jump
MF	Monitor flag
MOS	Metal oxide semiconductor
NEA	Normal exit address
P	Program address
PE	Parity error
PP	Peripheral processor
PPM	Peripheral processor memory
PPS	Peripheral processor subsystem
PPU	Peripheral processor unit
PSD	Program status designator
RAC	Reference address for CM
RAE	Reference address for ECS
RAL	Reference address for LCME
RAS	Reference address for CM
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 4-7.

MODEL 175 AND MODEL 176 DIFFERENCES

Model 175 and model 176 differences involve the instruction stack, central processor instructions, and the peripheral processor subsystem instructions. The following descriptions summarize the differences.

INSTRUCTION STACK DIFFERENCES

Models 175 and 176 each contain a 12-word instruction stack. The stack is filled two words ahead of the program address being executed.

In model 175, the stack is voided by an exchange, return jump, jump to the content of Bi plus K (02 instruction) or a branch (03 through 07 instructions) to a location not in the stack. The stack always contains sequential code.

In model 176, the stack isvoided by an exchange or a return jump. The stack can contain nonsequential code or duplicate entries.

CENTRAL PROCESSOR INSTRUCTION DIFFERENCES

The central processor instruction differences occur for shift, floating-add, divide, branch, and central exchange.

SHIFT (22 AND 23 INSTRUCTIONS)

Model 175 returns a plus zero when a negative number is right shifted more than 63 (decimal) places. Model 176 returns a minus zero when a negative number is right shifted by more than 63 (decimal) places.

Example: 23011

X1 = 4000 0000 0000 0000 0000

B1 = 000100

Model 175 Result:

X0 = 0000 0000 0000 0000 0000

Model 176 Result:

X0 = 7777 7777 7777 7777 7777

Model 175 checks bits 6 through 10 and ignores bits 11 through 16. Model 176 checks bits 6 through 11 of Bj for a shift count greater than 63 (decimal) and ignores bits 12 through 16.

Example: 23011

X1 = 2525 2525 2525 2525 2525

B1 = 004001

Model 175 Result:

X0 = 1252 5252 5252 5252 5252

Model 176 Result:

X0 = 0000 0000 0000 0000 0000

FLOATING-ADD (30 INSTRUCTION)

When the difference between the exponents is greater than 128 (decimal), model 175 extends the shifted sign bit to the entire shifted operand. Model 176 enters a shifted operand of plus zero regardless of the sign of the shifted operand. If the reference operand has a zero coefficient, the results can differ in sign.

Example: 30012

X1 = 4277 7777 7777 7777 7777

X2 = 5277 5555 5555 5555 5555

Model 175 Result:

X0 = 4277 7777 7777 7777 7777

Model 176 Result:

X0 = 3500 0000 0000 0000 0000

Reversing the operands (30021) gives the same results.

Example: 31012

X1 = 4277 7777 7777 7777 7777

X2 = 2500 2222 2222 2222 2222

Model 175 Result:

X0 = 4277 7777 7777 7777 7777

Model 176 Result:

X0 = 3500 0000 0000 0000 0000

Example: 31012

X1 = 5277 5555 5555 5555 5555

X2 = 3500 0000 0000 0000 0000

Model 175 Result:

X0 = 4277 7777 7777 7777 7777

Model 176 Result:

X0 = 3500 0000 0000 0000 0000

Reversing the operands (31021) on either of the examples for a floating difference gives compatible results for models 175 and 176. The result on both models is 3500 0000 0000 0000 0000.

DIVIDE (45 INSTRUCTION)

Model 175 adds one third to the dividend in a divide round. Model 176 adds one half to the dividend in a divide round.

Example: 45012

X1 = 2027 7223 2220 7175 5360

X2 = 1347 4255 6115 0364 7225

Model 175 Result:

X0 = 2430 6557 3505 0613 2700

Model 176 Result:

X0 = 2430 6557 3505 0613 2701

If the exponent subtract causes an underflow or overflow, model 175 returns an indefinite result with a divide fault. With the same conditions, model 176 returns an underflow or overflow result even with a divide fault.

Example: 44012

X1 = 3700 0222 0000 0000 0000

X2 = 1600 0022 0000 0000 0000

Model 175 Result:

X0 = 1777 0000 0000 0000 0000

Model 176 Result:

X0 = 3777 0000 0000 0000 0000

BRANCH (034 AND 035 INSTRUCTIONS)

In model 175, overflow is out of range. In model 176, overflow and indefinite are out of range.

CENTRAL EXCHANGE (013 INSTRUCTION)

The 013 instruction in model 175 causes an exchange jump to the content of Bj plus K. In model 176, the instruction causes an exchange jump to the content of Bj plus K plus the reference address for CM.

In model 175, the monitor flag controls the operation of the 013 instruction. The instruction changes the state of the monitor flag.

In model 176, the exit mode flag in the PSD register controls the operation of the 013 instruction. The exit mode flag sets to the value specified in the entering exchange package.

PERIPHERAL PROCESSOR SUBSYSTEM INSTRUCTION DIFFERENCES

The PPS instruction differences occur for exchange jump, monitor exchange jump, and monitor exchange jump to MA instructions.

EXCHANGE JUMP (260x INSTRUCTION

In model 175, the exchange jump initiates an exchange jump of the CP to the 18-bit address specified by the A register. In model 176, the instruction performs as a 261 instruction.

MONITOR EXCHANGE JUMP (261x INSTRUCTION)

In model 175, the monitor exchange jump instruction is enabled or disabled by the CEJ/MEJ switch. When the instruction is enabled and the monitor flag is clear, the instruction sets the monitor flag and initiates an exchange jump to the 18-bit address specified by the A register. If the monitor flag is set, the instruction acts as a pass instruction. When the instruction is disabled by the CEJ/MEJ switch, the instruction executes as a 260 instruction.

In model 176, the CEJ/MEJ switch has no function. A monitor exchange jump in this model initiates an exchange jump of the CP. The jump is to the 18-bit address specified by the A register. The jump occurs only if the exit mode flag is clear and no I/O interrupts are waiting to be processed. If the exit mode flag is set or I/O interrupts are waiting to be processed, the instruction acts as a pass instruction.

MONITOR EXCHANGE JUMP TO MA (262x INSTRUCTION)

In model 175, the monitor exchange jump to MA instruction is enabled or disabled by the CEJ/MEJ switch. When the instruction is enabled and the monitor flag is clear, the instruction sets the flag and initiates an exchange jump to the 18-bit address specified by the MA register. If the monitor flag is set, the instruction acts as a pass instruction. When the instruction is disabled by the CEJ/MEJ switch, the instruction executes as a 260 instruction.

In model 176, the CEJ/MEJ switch has no function. A monitor exchange jump to MA instruction performs as a 261 instruction.

INDEX

COMMENT SHEET

MANUAL TITLE CDC CYBER 170 Computer Systems
Hardware Reference Manual

PUBLICATION NO. 60420000 REVISION H

FROM:

NAME:

BUSINESS ADDRESS:

CUT ALONG LINE

AA3419 REV. 7/75 PRINTED IN U.S.A.

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