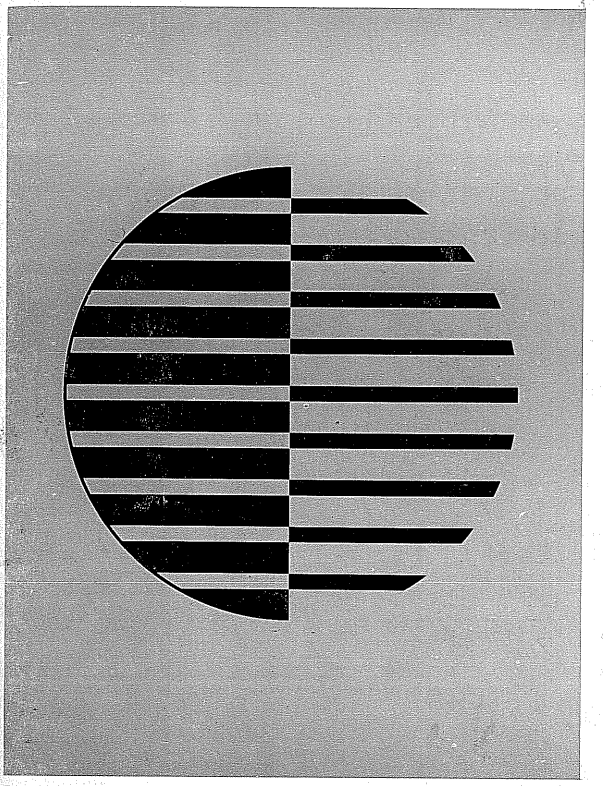
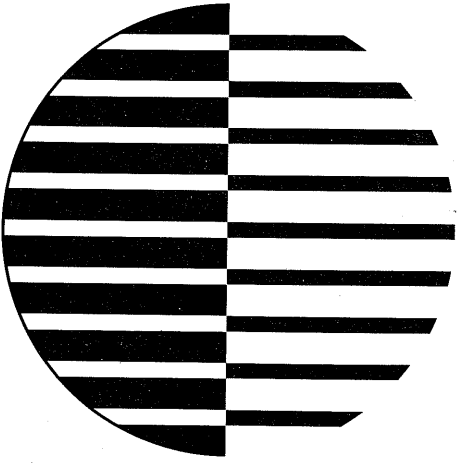
CODES/CONTROL DATA 6600
Computer System



CODES/CONTROL DATA® 6600 Computer System



Record of Revisions

REVISION	NOTES
A	System Macros added. Obsoletes all previous editions.
(4-30-65)	

Pub. No. 60141900
May, 1965

© 1965, Control Data Corporation
Printed in the United States of America

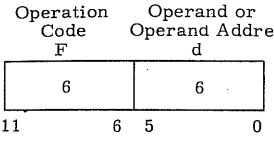
Address comments concerning this manual to:

Control Data Corporation
Technical Publications Department
4201 North Lexington Avenue
St. Paul, Minnesota 55112

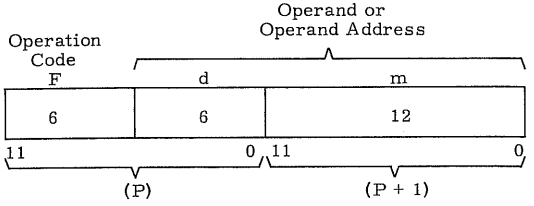
CONTENTS

EXPLANATION OF PERIPHERAL AND CONTROL PROCESSOR INSTRUCTION FORMATS

An instruction may have a 12-bit or a 24-bit format. 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, which is the contents of the next program address (P + 1), with d to form an 18-bit operand or operand address.



EXPLANATION OF SYMBOLS USED IN PERIPHERAL AND CONTROL PROCESSOR INSTRUCTION LISTINGS

d	Implies d itself
(d)	Implies the contents of d
((d))	Implies the contents of the location specified by d
m	Implies m itself used as an address
m + (d)	The contents of d are added to m to form an operand (jump address)
(m + (d))	The contents of d are added to m to form the address of the operand
dm	Implies an 18-bit quantity with d as the upper 6 bits and m as the lower 12 bits

PERIPHERAL AND CONTROL PROCESSOR INSTRUCTIONS

1. NUMERICAL LISTING

F	MNE-MONIC	AD-DRESS	NAME	TIME (Major Cycles)
00	PSN		Pass	1
01	LJM	m d	Long jump to m + (d)	2-3
02	RJM	m d	Return jump to m + (d)	3-4
03	UJN	d	Unconditional jump d	1
04	ZJN	d	Zero jump d	1
05	NJN	d	Nonzero jump d	1
06	PJN	d	Plus jump d	1
07	MJN	d	Minus jump d	1
10	SHN	d	Shift d	1
11	LMN	d	Logical difference d	1
12	LPN	d	Logical product d	1
13	SCN	d	Selective clear d	1
14	LDN	d	Load d	1
15	LCN	d	Load complement d	1
16	ADN	d	Add d	1
17	SBN	d	Subtract d	1
20	LDC	dm	Load dm	2
21	ADC	dm	Add dm	2
22	LPC	dm	Logical product dm	2
23	LMC	dm	Logical difference dm	2
24	PSN		Pass	1
25	PSN		Pass	1
26	EXN		Exchange jump	min. 2
27	RPN		Read program address	1
30	LDD	d	Load (d)	2
31	ADD	d	Add (d)	2
32	SBD	d	Subtract (d)	2
33	LMD	d	Logical difference (d)	2
34	STD	d	Store (d)	2

F	MNE-MONIC	AD-DRESS	NAME	TIME (Major Cycles)
35	RAD	d	Replace add (d)	3
36	AOD	d	Replace add one (d)	3
37	SOD	d	Replace subtract one (d)	3
40	LDI	d	Load ((d))	3
41	ADI	d	Add ((d))	3
42	SBI	d	Subtract ((d))	3
43	LMI	d	Logical difference ((d))	3
44	STI	d	Store ((d))	3
45	RAI	d	Replace add ((d))	4
46	AOI	d	Replace add one ((d))	4
47	SOI	d	Replace subtract one ((d))	4
50	LDM	m d	Load (m + (d))	3-4
51	ADM	m d	Add (m + (d))	3-4
52	SBM	m d	Subtract (m+(d))	3-4
53	LMM	m d	Logical difference (m + (d))	3-4
54	STM	m d	Store (m+(d))	3-4
55	RAM	m d	Replace add (m + (d))	4-5
56	AOM	m d	Replace add one (m + (d))	4-5
57	SOM	m d	Replace subtract one (m + (d))	4-5
60	CRD	d	Central read from (a) to d	min.6
61	CRM	m d	Central read (d) words from (A) to m	5 plus 5/word
62	CWD	d	Central write to (A) from d	min.6
63	CWM	m d	Central write (d) words to (A) from m	5 plus 5/word
64	AJM	m d	Jump to m if channel d active	2
65	IJM	m d	Jump to m if channel d inactive	2

F	MNE-MONIC	AD-DRESS	NAME	TIME (Major Cycles)
66	FJM	m d	Jump to m if channel d full	2
67	EJM	m d	Jump to m if channel d empty	2
70	IAN	d	Input to A from channel d	2
71	IAM	m d	Input (A) words to m from channel d	4 plus 1/word
72	OAN	d	Output from A on channel d	2
73	OAM	m d	Output (A) words from m on channel d	4 plus 1/word
74	ACN	d	Activate channel d	2
75	DCN	d	Disconnect channel d	2
76	FAN	d	Function (A) on channel d	2
77	FNC	m d	Function m on channel d	2

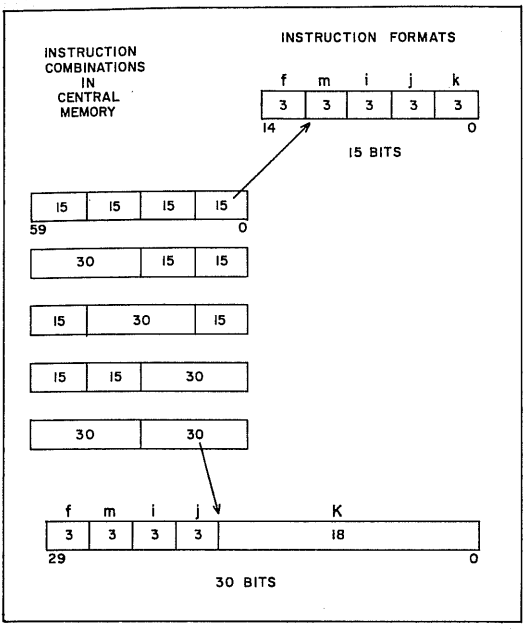
2. ALPHABETICAL LISTING

F	MNE-MONIC	AD-DRESS-	NAME	TIME (Major Cycles)
74	ACN	d	Activate channel d	2
21	ADC	dm	Add dm	2
31	ADD	d	Add (d)	2
41	ADI	d	Add ((d))	3
51	ADM	m d	Add (m + (d))	3-4
16	ADN	d	Add d	1
64	AJM	m d	Jump to m if channel d active	2
36	AOD	d	Replace add one (d)	3
46	AOI	d	Replace add one ((d))	4
56	AOM	m d	Replace add one (m + (d))	4-5
60	CRD	d	Central read from (A) to d	min. 6
61	CRM	m d	Central read (d) words from (A) to m	5 plus 5/word
62	CWD	d	Central write to (A) from d	min. 6
63	CWM	m d	Central write (d) words to (A) from m	5 plus 5/word
75	DCN	d	Disconnect channel d	2
67	EJM	m d	Jump to m if channel d empty	2
26	EXN		Exchange jump	min. 2
76	FAN	d	Function (A) on channel d	2
66	FJM	m d	Jump to m if channel d full	2
77	FNC	m d	Function m on channel d	2
71	IAM	m d	Input (A) words to m from channel d	4 plus 1/word
70	IAN	d	Input to A from channel d	2

F	MNE-MONIC	AD-DRESS	NAME	TIME (Major Cycles)
65	IJM	m d	Jump to m if channel d inactive	2
15	LCN	d	Load complement d	1
20	LDC	dm	Load dm	2
30	LDD	d	Load (d)	2
40	LDI	d	Load ((d))	3
50	LDM	m d	Load (m + (d))	3-4
14	LDN	d	Load d	1
01	LJM	m d	Long jump to m + (d)	2-3
23	LMC	dm	Logical difference dm	2
33	LMD	d	Logical difference (d)	2
43	LMI	d	Logical difference ((d))	3
53	LMM	m d	Logical difference (m+(d))	3-4
11	LMN	d	Logical difference d	1
22	LPC	dm	Logical product dm	2
12	LPN	d	Logical product d	1
07	MJN	d	Minus jump d	1
05	NJN	d	Nonzero jump d	1
73	OAM	m d	Output (A) words from m on channel d	4 plus 1/word
72	OAN	d	Output from A on channel d	2
06	PJN	d	Plus jump d	1
00	PSN		Pass	1
24	PSN		Pass	1
25	PSN		Pass	1
35	RAD	d	Replace add (d)	3

F	MNE-MONIC	AD-DRESS	NAME	TIME (Major Cycles)
45	RAI	d	Replace add ((d))	4
55	RAM	m d	Replace add (m + (d))	4-5
02	RJM	m d	Return jump to m + (d)	3-4
27	RPN		Read program address	1
32	SBD	d	Subtract (d)	2
42	SBI	d	Subtract ((d))	3
52	SBM	m d	Subtract (m+(d))	3-4
17	SBN	d	Subtract d	1
13	SCN	d	Selective clear d	1
10	SHN	d	Shift d	1
37	SOD	d	Replace subtract one (d)	3
47	SOI	d	Replace subtract one ((d))	4
57	SOM	m d	Replace subtract one (m + (d))	4-5
34	STD	d	Store (d)	2
44	STI	d	Store ((d))	3
54	STM	m d	Store (m + (d))	3-4
03	UJN	d	Unconditional jump d	1
04	ZJN	d	Zero jump d	1

EXPLANATION OF CENTRAL PROCESSOR INSTRUCTION FORMATS



EXPLANATION OF SYMBOLS USED IN CENTRAL PROCESSOR INSTRUCTION LISTINGS

A	One of eight address registers (18 bits)
B	One of eight index registers (18 bits) B0 is fixed and equal to zero
fm	Instruction code (6 bits)
i	Specifies which of eight designated registers (3 bits). Is also used in 03X instructions as part of a 9-bit operation code.
j	Specifies which of eight designated registers (3 bits)
jk	Constant, indicating number of shifts to be taken (6 bits)
k	Specifies which of eight designated registers (3 bits)
K	Constant, indicating branch destination or operand (18 bits)
X	One of eight operand registers (60 bits)

CENTRAL PROCESSOR INSTRUCTIONS

1. NUMERICAL LISTING

BRANCH UNIT

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
00	PS		Program stop	-
01	RJ	K	Return jump to K	13
02	JP	Bi + K	Jump to Bi + K	14*
030	ZR	Xj K	Jump to K if Xj = 0	9*
031	NZ	Xj K	Jump to K if Xj ≠ 0	9*
032	PL	Xj K	Jump to K if Xj = plus (positive)	9*
033	NG	Xj K	Jump to K if Xj = negative	9*
034	IR	Xj K	Jump to K if Xj is in range	9*
035	OR	Xj K	Jump to K if Xj is out of range	9*
036	DF	Xj K	Jump to K if Xj is definite	9*
037	ID	Xj K	Jump to K if Xj is indefinite	9*
04	EQ	Bi Bj K	Jump to K if Bi = Bj	8*
04	ZR	Bi K	Jump to K if Bi = B0	8*
05	NE	Bi Bj K	Jump to K if Bi ≠ Bj	8*

Note 1. GO TO K + Bi and GO TO K if Bi---tests made in increment unit.

Note 2. GO TO K if Xj---tests made in long add unit.

*Add 6 minor cycles to branch time for a branch to an instruction which is out of the stack (no memory conflict considered). Add 2 minor cycles to branch time for a no branch condition in the stack. Add 5 minor cycles to branch time for a no branch condition out of the stack.

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
05	NZ	Bi K	Jump to K if Bi ≠ B0	8*
06	GE	Bi Bj K	Jump to K if Bi ≧ Bj	8*
06	PL	Bi K	Jump to K if Bi ≧ B0	8*
07	LT	Bi Bj K	Jump to K if Bi < Bj	8*
07	NG	Bi K	Jump to K if Bi < B0	8*
BOOLEAN UNIT				
10	BXi	Xj	Transmit Xj to Xi	3
11	BXi	Xj*Xk	Logical Product of Xj & Xk to Xi	3
12	BXi	Xj + Xk	Logical sum of Xj & Xk to Xi	3
13	BXi	Xj - Xk	Logical difference of Xj & Xk to Xi	3
14	BXi	-Xk	Transmit the comp. of Xk to Xi	3
15	BXi	-Xk*Xj	Logical product of Xj & Xk comp. to Xi	3
16	BXi	-Xk + Xj	Logical sum of Xj & Xk comp of Xi	3
17	BXi	-Xk - Xj	Logical difference of Xj & Xk comp. to Xi	3

*Add 6 minor cycles to branch time for a branch to an instruction which is out of the stack (no memory conflict considered). Add 2 minor cycles to branch time for a no branch condition in the stack. Add 5 minor cycles to branch time for a no branch condition out of the stack.

SHIFT UNIT

fm (i)	MNE-MONIC	AD-DRESS		NAME	TIME (Minor Cycles)
20	LXi	jk		Left shift Xi, jk places	3
21	AXi	jk		Arithmetic right shift Xi, jk places	3
22	LXi	Bj	Xk	Left shift Xk nominally Bj places to Xi	3
23	AXi	Bj	Xk	Arithmetic right shift Xk nominally Bj places to Xi	3
24	NXi	Bj	Xk	Normalize Xk in Xi and Bj	4
25	ZXi	Bj	Xk	Round and normalize Xk in Xi and Bj	4
26	UXi	Bj	Xk	Unpack Xk to Xi and Bj	3
27	PXi	Bj	Xk	Pack Xi from Xk and Bj	3
43	MXi	jk		Form mask in Xi, jk bits	3

ADD UNIT

30	FXi	Xj + Xk		Floating sum of Xk and Xk to Xi	4
31	FXi	Xj - Xk		Floating difference Xj and Xk to Xi	4
32	DXi	Xj + Xk		Floating DP sum of Xj and Xk to Xi	4
33	DXi	Xj - Xk		Floating DP difference of Xj and Xk to Xi	4
34	RXi	Xj + Xk		Round floating sum of Xj and Xk to Xi	4
35	RXi	Xj - Xk		Round floating difference of Xj and Xk to Xi	4

LONG ADD UNIT

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
36	IXi	Xj + Xk	Integer sum of Xj and Xk to Xi	3
37	IXi	Xj - Xk	Integer difference of Xj and Xk to Xi	3

MULTIPLY UNIT

40	FXi	Xj * Xk	Floating product of Xj and Xk to Xi	10
41	RXi	Xj * Xk	Round floating product of Xj & Xk to Xi	10
42	DXi	Xj * Xk	Floating DP product of Xj & Xk to Xi	10

DIVIDE UNIT

44	FXi	Xj / Xk	Floating divide Xj by Xk to Xi	29
45	RXi	Xj / Xk	Round floating divide Xj by Xk to Xi	29
46	NO		No operation	-
47	CXi	Xk	Count the number of 1's in Xk to Xi	8

INCREMENT UNIT

50	SAi	Aj + K	Set Ai to Aj + K	3
50	SAi	Aj - K	Set Ai to Aj + comp. of K	3
51	SAi	Bj + K	Set Ai to Bj + K	3
51	SAi	Bj - K	Set Ai to Bj + comp. of K	3
52	SAi	Xj + K	Set Ai to Xj + K	3
52	SAi	Xj - K	Set Ai to Xj + comp. of K	3
53	SAi	Xj + Bk	Set Ai to Xj + Bk	3
54	SAi	Aj + Bk	Set Ai to Aj + Bk	3
55	SAi	Aj - Bk	Set Ai to Aj - Bk	3

INCREMENT UNIT (Continued)

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
56	SAi	Bj + Bk	Set Ai to Bj + Bk	3
57	SAi	Bj - Bk	Set Ai to Bj - Bk	3
60	SBi	Aj + K	Set Bi to Aj + K	3
60	SBi	Aj - K	Set Bi to Aj + comp. of K	3
61	SBi	Bj + K	Set Bi to Bj + K	3
61	SBi	Bj - K	Set Bi to Bj + comp. of K	3
62	SBi	Xj + K	Set Bi to Xj + K	3
62	SBi	Xj - K	Set Bi to Xj + comp. of K	3
63	SBi	Aj + Bk	Set Bi to Xj + Bk	3
64	SBi	Aj + Bk	Set Bi to Aj + Bk	3
65	SBi	Aj - Bk	Set Bi to Aj - Bk	3
66	SBi	Bj + Bk	Set Bi to Bj + Bk	3
67	SBi	Bj - Bk	Set Bi to Bj - Bk	3
70	SXi	Aj + K	Set Xi to Aj+ K	3
70	SXi	Aj - K	Set Xi to Aj + comp. of K	3
71	SXi	Bj + K	Set Xi to Bj + K	3
71	SXi	Bj - K	Set Xi to Bj + comp. of K	3
72	SXi	Xj + K	Set Xi to Xj + K	3
72	SXi	Xj - K	Set Xi to Xj + comp. of K	3
73	SXi	Xj + Bk	Set Xi to Xj + Bk	3
74	SXi	Aj + Bk	Set Xi to Aj + Bk	3
75	SXi	Aj - Bk	Set Xi to Aj - Bk	3
76	SXi	Bj + Bk	Set Xi to Bj + Bk	3
77	SXi	Bj - Bk	Set Xi to Bj - Bk	3

2. ALPHABETICAL LISTING

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
21	AXi	jk	Arithmetic right shift Xi, jk places	3
23	AXi	Bj Xk	Arithmetic right shift Xk nominally Bj places to Xi	3
10	BXi	Xj	Transmit Xj to Xi	3
11	BXi	Xj*Xk	Logical product of Xj & Xk to Xi	3
12	BXi	Xj + Xk	Logical sum of Xj & Xk to Xi	3
13	BXi	Xj - Xk	Logical difference of Xj & Xk to Xi	3
14	BXi	-Xk	Transmit the comp. of Xk to Xi	3
15	BXi	-Xk*Xj	Logical product of Xj & Xk comp. to Xi	3
16	BXi	-Xk + Xj	Logical sum of Xj & Xk comp. to Xi	3
17	BXi	-Xk -Xj	Logical difference of Xj & Xk comp. to Xi	3
47	CXi	Xk	Count the number of 1's in Xk to Xi	8
036	DF	Xj K	Jump to K if Xj is definite	9*
32	DXi	Xj + Xk	Floating DP sum of Xj and Xk to Xi	4
33	DXi	Xj - Xk	Floating DP difference of Xj and Xk to Xi	4
42	DXi	Xj * Xk	Floating DP product of Xj & Xk to Xi	10

*Add 6 minor cycles to branch time for a branch to an instruction which is out of the stack (no memory conflict considered). Add 2 minor cycles to branch time for a no branch condition in the stack. Add 5 minor cycles to branch time for a no branch condition out of the stack.

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
04	EQ	Bi Bj K	Jump to K if Bi = Bj	8*
30	FXi	Xj + Xk	Floating sum of Xj and Xk to Xi	4
31	FXi	Xj - Xk	Floating difference Xj and Xk to Xi	4
40	FXi	Xj *Xk	Floating product of Xj and Xk to Xi	10
44	FXi	Xj / Xk	Floating divide Xj by Xk to Xi	29
06	GE	Bi Bj K	Jump to K if Bi ≧ Bj	8*
037	ID	Xj K	Jump to K if Xj is indefinite	9*
034	IR	Xj K	Jump to K if Xj is in range	9*
36	IXi	Xj + Xk	Integer sum of Xj and Xk to Xi	3
37	IXi	Xj - Xk	Integer difference of Xj and Xk to Xi	3
02	JP	Bi + K	Jump to Bi + K	14*
07	LT	Bi Bj K	Jump to K if Bi < Bj	8*
20	LXi	jk	Left shift Xi, jk places	3
22	LXi	Bj Xk	Left shift Xk nominally Bj places to Xi	3
43	MXi	jk	Form mask in Xi, jk bits	3

*Add 6 minor cycles to branch time for a branch to an instruction which is out of the stack (no memory conflict considered). Add 2 minor cycles to branch time for a no branch condition in the stack. Add 5 minor cycles to branch time for a no branch condition out of the stack.

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
05	NE	Bi Bj K	Jump to K if Bi ≠ Bj	8*
033	NG	Xj K	Jump to K if Xj = negative	9*
07	NG	Bi K	Jump to K if Bi < B0	8*
46	NO		No operation	-
24	NXi	Bj Xk	Normalize Xk in Xi and Bj	4
031	NZ	Xj K	Jump to K if Xj ≠ 0	9*
05	NZ	Bi K	Jump to K if Bi ≠ B0	8*
035	OR	Xj K	Jump to K if Xj is out of range	9*
032	PL	Xj K	Jump to K if Xj = plus (positive)	9*
06	PL	Bi K	Jump to K if Bi ≧ B0	8*
00	PS		Program stop	-
27	PXi	Bj Xk	Pack Xi from Xk and Bj	3
01	RJ	K	Return jump to K	13
34	RXi	Xj + Xk	Round floating sum of Xj and Xk to Xi	4
35	RXi	Xj - Xk	Round floating difference of Xj and Xk to Xi	4
41	RXi	Xj * Xk	Round floating product of Xj & Xk to Xi	10

*Add 6 minor cycles to branch time for a branch to an instruction which is out of the stack (no memory conflict considered). Add 2 minor cycles to branch time for a no branch condition in the stack. Add 5 minor cycles to branch time for a no branch condition out of the stack.

fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
45	RXi	Xj / Xk	Round floating divide Xj by Xk to Xi	29
50	SAi	Aj + K	Set Ai to Aj + K	3
50	SAi	Aj - K	Set Ai to Aj + comp. of K	3
51	SAi	Bj + K	Set Ai to Bj + K	3
51	SAi	Bj - K	Set Ai to Bj + comp. of K	3
52	SAi	Xj + K	Set Ai to Xj + K	3
52	SAi	Xj - K	Set Ai to Xj + comp. of K	3
53	SAi	Xj + Bk	Set Ai to Xj + Bk	3
54	SAi	Aj + Bk	Set Ai to Aj + Bk	3
55	SAi	Aj - Bk	Set Ai to Aj - Bk	3
56	SAi	Bj + Bk	Set Ai to Bj + Bk	3
57	SAi	Bj - Bk	Set Ai to Bj - Bk	3
60	SBi	Aj + K	Set Bi to Aj + K	3
60	SBi	Aj - K	Set Bi to Aj + comp. of K	3
61	SBi	Bj + K	Set Bi to Bj + K	3
61	SBi	Bj - K	Set Bi to Bj + comp. of K	3
62	SBi	Xj + K	Set Bi to Xj + K	3
62	SBi	Xj - K	Set Bi to Xj + comp. of K	3
63	SBi	Xj + Bk	Set Bi to Xj + Bk	3
64	SBi	Aj + Bk	Set Bi to Aj + Bk	3
65	SBi	Aj - Bk	Set Bi to Aj - Bk	3
66	SBi	Bj + Bk	Set Bi to Bj + Bk	3
67	SBi	Bj - Bk	Set Bi to Bj - Bk	3
70	SXi	Aj + K	Set Xi to Aj + K	3
70	SXi	Aj - K	Set Xi to Aj + comp. of K	3
71	SXi	Bj + K	Set Xi to Bj + K	3
71	SXi	Bj - K	Set Xi to Bj + comp. of K	3

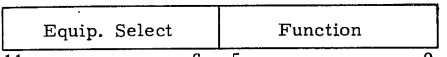
fm (i)	MNE-MONIC	AD-DRESS	NAME	TIME (Minor Cycles)
72	SXi	Xj + K	Set Xi to Xk + K	3
72	SXi	Xj - K	Set Xi to Xj + comp. of K	3
73	SXi	Xj + Bk	Set Xi to Xj + Bk	3
74	SXi	Aj + Bk	Set Xi to Aj + Bk	3
75	SXi	Aj - Bk	Set Xi to Aj - Bk	3
76	SXi	Bj + Bk	Set Xi to Bj + Bk	3
77	SXi	Bj - Bk	Set Xi to Bj - Bk	3
26	UXi	Bj Xk	Unpack Xk to Xi and Bj	3
030	ZR	Xj K	Jump to K if Xj = 0	8*
04	ZR	Bi K	Jump to K if Bi = B0	8*
25	ZXi	Bj Xk	Round and normalize Xk in Xi and Bj	4

*Add 6 minor cycles to branch time for a branch to an instruction which is out of the stack (no memory conflict considered). Add 2 minor cycles to branch time for a no branch condition in the stack. Add 5 minor cycles to branch time for a no branch condition out of the stack.

EXTERNAL FUNCTION CODES AND STATUS RESPONSES

1. 405-B CARD READER

Function Word

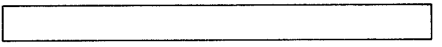


0700	De-select
0701	Gate Card to Secondary Bin
0702	Read Non-Stop
0704	Status Request

Status Reply: 0000 = Ready
0001 = Not Ready
0002 = End of File
0004 = Compare Error

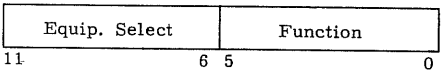
To read one card, execute successive S702 and S704 functions.

Data Word



2. 415-B, 523, & 544 CARD PUNCHES

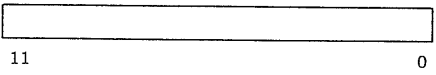
Function Word



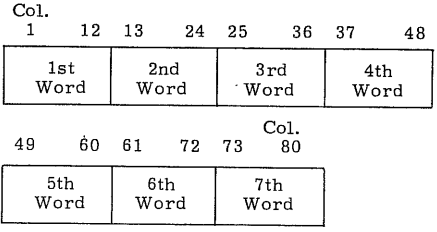
3000	De-select
3001	Offset *
3002	Select Punch
3040	Status Request

Status Reply: 0000 = Ready
0200 = Manual Switch in 1604 Position
2000 = Not Ready

Data Word



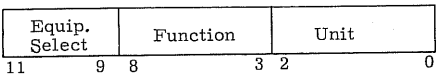
Word Assembly Sequence



* 415-B only

3. 607-B MAGNETIC TAPE UNIT

Function Word

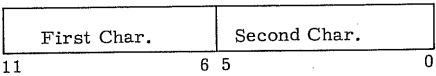


200x	Select
201x	Write Binary
202x	Read Binary
203x	Backspace
206x	Rewind
207x	Rewind Unload
210x	Status Request
221x	Write BCD
222x	Read BCD
261x	Write File Mark

Status Reply: 0X00 = Ready
0X01 = Not Ready
0X02 = Parity Error
0X04 = Load Point
0X10 = End of Tape
0X20 = File Mark
0X40 = Write Lockout

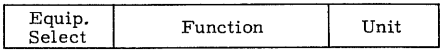
X = 0: 800 bpi
X = 1: 556 bpi
X = 2: 200 bpi

Data Word



4. 6622 MAGNETIC TAPE CONTROLLER (626-B MAGNETIC TAPE UNIT)

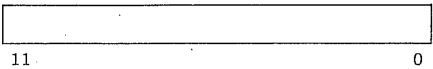
Function Word



300x	Selcct	(x = 0-7)
301x	Write Binary	
302x	Read Binary	
303x	Backspace	
306x	Rewind	
307x	Rewind Unload	
310x	Status Request	
361x	Write File Mark	

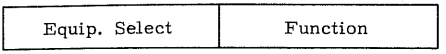
Status Reply:
0000 = Ready
0001 = Not Ready
0002 = Parity Error
0004 = Load Point
0010 = End of Tape
0020 = File Mark
0040 = Write Lockout

Data Word



5. 1612 PRINTER

Function Word



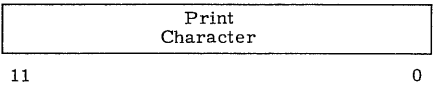
A. FUNCTION CODES

0600	Select Printer
0601	Single Space
0602	Double Space
0603	Move Paper to Format Channel 7
0604	Move Paper to Top of Form
0605	Print
0606	Suppress Line Advance After Next Print
0607	Status Request
0610	Clear Format Channels
061X	Select Format Channel X (X = 1-6)

B. STATUS REPLY

0000	Not Ready
4000	Ready

Data Word



6. 1612 FUNCTION DESCRIPTION

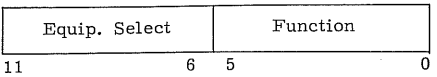
Function codes 0601, 0602, 0603 and 0604 are executed on receipt. One advance is carried out each time the Function code is received.

The printer automatically advances one line after each 0605 Function code unless the 0605 is preceded by an 0606 Function code.

Format channel selections (061X) DO NOT initiate paper movement. They control how far paper moves after an 0605 Function code. A selection remains active until a Clear Format Channel (0610) Function code is received.

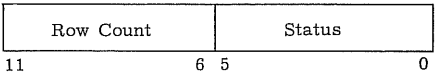
7. 501-B PRINTER

Function Word



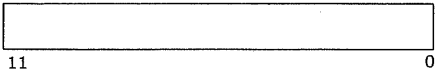
0600	Select Print
0601	Advance Paper
0603	Advance to Selected Line (Channel 7)
0604	Top of Form (Channel 8)
0606	End of Data
0607	Select Status Request
0610	Clear Channel Selections
061X	Select Format Channel X (X = 1 - 6)

Status Reply Word



XXX1	Not Index Mark
XXX2	Wait Character Mark
XXX4	Holding Row Count
XX1X	Paper Advancing
XX2X	Not Ready

Data Word



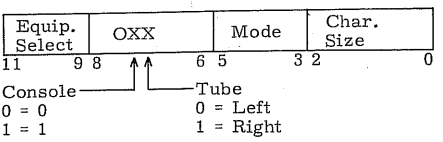
8. 501-B FUNCTION DESCRIPTION

The 501-B does not have a one-line memory. A 136-bit Holding register is used to hold printing information. For a line of print, the Holding register must be loaded once for each character on the print drum. A "1" bit must be loaded in the Holding register in each column in which a given character is to be printed. If a character is not used, at least one word of zeroes must be sent to the Holding register. Printing starts with the number 0 and proceeds in the order in which the characters appear on the print drum. After all characters in a line have been printed, an End of Data code must be sent.

Paper advance is not automatic. An Advance Paper Code (S601) must be sent after each line is printed. If no format channel is selected, paper will advance one line. If a format channel is selected, paper will advance as dictated by that channel.

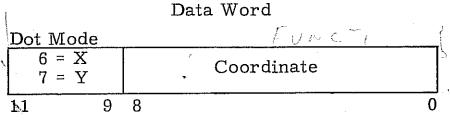
9. 6602 CONSOLE DISPLAY

Function Word

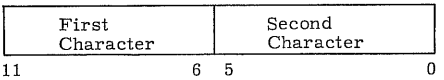


Console 0	Console 1	
7000	7200	Select 64 Char./Line, Left Screen
7001	7201	Select 32 Char./Line, Left Screen
7002	7202	Select 16 Char./Line, Left Screen
7010	7210	Select 512 Dots/Line
7020	7220	Select Keyboard Input
7100	7300	Select 64 Char./Line, Right Screen
7101	7301	Select 32 Char./Line, Right Screen
7102	7302	Select 16 Char./Line, Right Screen

Data Word

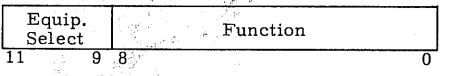


Character Mode

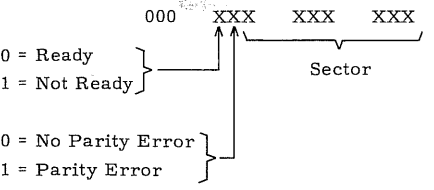


10. 6603 DISK SYSTEM

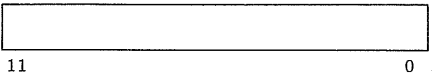
Function Word



10xx - 11xx	Read Sector X
12xx - 13xx	Write Sector X
14xx - 15xx	Select Track X
160x	Select Head Group X
1700	Status Request



Data Word



11. 6681 DATA CHANNEL CONVERTER (3000 SERIES INTERFACE)

Equipment Select

11 0

2000 Select
2100 Deselect

Mode I Function Word

	Function

8 0

Connect:	4XXX	Connect External Equipment 4
	5XXX	Connect External Equipment 5
	6XXX	Connect External Equipment 6
	7XXX	Connect External Equipment 7
Function:	0XXX	Send 9-bit Function Code to External Equipment Connected

Mode II Function Word

Function

11 0

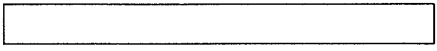
Connect:	1000	Select 6681 to Output a 12-bit Connect Code
Function:	1100	Select 6681 to Output a 12-bit Function Code to External Equipment Connected

Status	1200	6681 Status Request – STAT. A	
	1300	External Equipment Status Request STAT B	
Status Reply:	XXX1	Reject (internal or external)	
	XXX2	Internal Reject	
	XXX4	Transmission Parity Error	
	XX1X - 2XXX	Eight Interrupt Lines	

SHOULD BE LAST INSTR BEFORE ACTUAL INPUT INSTR FROM PP. DUE TO CLR - SET INPUT ON FUNCT.

Data I/O:	1400	Input to End of Record
	1500	Input until 6600 Sends Inactive Signal
	1600	Output until 6600 Sends Inactive Signal
	17XY	EXTERNAL M/C ON SELECTED EQUIP.

Data Word



SYSTEM MACROS

1. MAGNETIC TAPE OPERATIONS

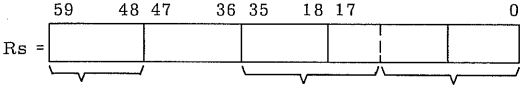
Opcode	Address Field	Remarks*
RQTW	N, S	Request tape assignment from system.
DRTW	N, S	Release tape back to system.
SFFW	N, S	Search file mark forward.
SFBW	N, S	Search file mark backward
WFMW	N, S	Write file mark
RWLW	N, S	Rewind tape to load point.
RWUW	N, S	Rewind tape for unload.
FSPW	N, S, K	Forespace
BSPW	N, S, K	Backspace
RFCW	N, S, BA, EA RL, C	Read tape forward coded mode
RFBW	N, S, BA, EA, RL, C	Read tape forward binary mode.
WRCW	N, S, BA, EA, RL, C	Write tape coded mode.
WRBW	N, S, BA, EA, RL, C	Write tape binary mode.

*Wait if W is used.

N = Magnetic tape logical unit number; 1, 2, . . . M for M tape units in the system.
S = Location containing central memory address for status response code from System Peripheral Processor I/O Routine.
K = Number of logical tape records.
BA = Location containing beginning address of buffer area in central memory.
EA = Location containing ending address + 1 of buffer area in central memory.
RL = Number of 60-bit words per tape record
C = Conversion mode.
Blank or 0 - No Conversion
1 - BCD to display code.
2 - Display code to BCD.

STATUS RESPONSE CODES - positioned as per address S.

Rs = 0 Request completed with no trouble.
Rs = 1 Request in process.
Rs < 0 Request aborted. Reason given in bits 58 - 48.



No. of words in record where read length occurred.

No. of records completed including bad one.

Program error - BA > EA. (BIT 48)

End of file. (BIT 49)

Read length error. (BIT 51)

Write parity error unrecoverable. (BIT 52)

Read parity error unrecoverable. (BIT 53)

End of tape mark encountered before function completed (forward). (BIT 54)

Load point encountered before function completed (backward). (BIT 55)

Device unassigned. (BIT 57)

Device not ready. (BIT 58)

Request aborted. (BIT 59)

where: 1 implies the condition exists.

0 implies the condition does not exist.

2. DISK TRANSFERS

Provision is made in the operating system for the programmer to read and write scratch data to and from disk storage units. Data are usually broken up into related blocks called files. The files, in turn, are segmented into the blocks of data that are transmitted at one time. These are called logical records. For most efficient utilization of disk storage, logical records contain a minimum of 512 central memory words. A file is defined by an instruction or statement which specifies the number of 60-bit words in the longest record, the maximum number of logical records into which the file is to be segmented, and the symbolic name by which the file is identified. The actual data transmission is accomplished through the use of the following macro operators.

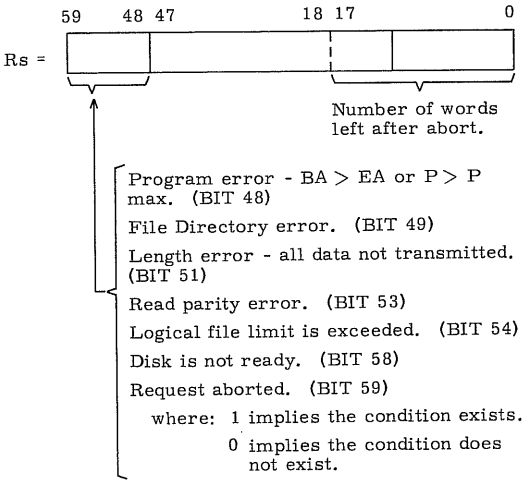
Opcode	Address Field	Remarks*
RDHW	N, S, BA, EA, NAME, P	Read record and hold data on disk.
RDRW	N, S, BA, EA, NAME, P	Read record and release data on disk.
WRDW	N, S, BA, EA, NAME, P	Write record on disk.

*Wait if W is used.

N = Disk logical unit number; 1, 2, ... M for M disk units in the system.
S = Location containing central memory address for status response code from System PP I/O routine.
BA = Location containing beginning address of buffer area in central memory.
EA = Location containing ending address + 1 of buffer area in central memory.
NAME = Symbolic name to identify disk logical file to be referenced.
P = Logical record number used to identify record read from disk or written onto disk.

STATUS RESPONSE CODES - positioned as per address S.

Rs = 0 Request is completed with no trouble.
Rs = 1 Request is in process.
Rs < 0 Request aborted. Reason given in bits 58 - 48.



3. PRINTER OPERATIONS

Opcode	Address Field	Remarks*
SSPW	N, S	Single space printer.
DSPW	N, S	Double space printer.
FC7W	N, S	Select Format Channel 7.
FC8W	N, S	Select Format Channel 8.
MC1W	N, S	Select Monitor Channel 1.
MC2W	N, S	Select Monitor Channel 2.
MC3W	N, S	Select Monitor Channel 3.
MC4W	N, S	Select Monitor Channel 4.
MC5W	N, S	Select Monitor Channel 5.
MC6W	N, S	Select Monitor Channel 6.
CMCW	N, S	Clear Monitor Channels 1-6.
SPAW	N, S	Suppress space after next print.
PRNW	N, S, BA, EA, RL, C	Print single line or multiple lines. **

*Wait if W is used.

**If SPA is given preceding a multiple line print, it applies only to the first line.

N = Printer logical unit number; 1, 2... M for M printers in the system.

S = Location containing central memory address for status response code from System Periph. Processor I/O Routine.

BA = Location containing beginning address of buffer area in central memory.

EA = Location containing ending address + 1 of buffer area in central memory.

RL = Number of 10 character words per line to print.

C = Conversion mode.

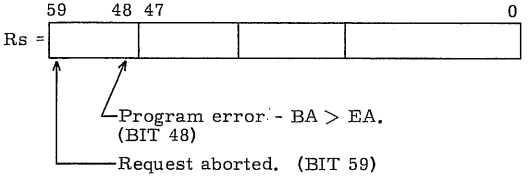
Blank or 0 - No conversion
2 - Display code to BCD.

STATUS RESPONSE CODES - positioned as per address S.

Rs = 0 Request is completed with no trouble.

Rs = 1 Request is in process.

Rs < 0 Request aborted. Reason given in bits 58 - 48.



where: 1 implies the condition exists.

0 implies the condition does not exist.

4. CARD OPERATIONS

Opcode	Address Field	Remarks*
PCHW	N, S, BA, EA, RL, C	Punch cards.
RDCW	N, S, BA, EA, RL, C	Read cards.

*Wait if W is used.

N = Card reader or punch logical unit number; 1, 2, ... M for M readers or punches in the system.

S = Location containing central memory address for status response code from System Periph. Processor I/O Routine.

BA = Location containing beginning address of buffer area in central memory.

EA = Location containing ending address +1 of buffer area in central memory.

RL = Number of leftmost 10-character fields or 5 columns of the card.

C = Conversion mode.

Blank or 0 - No conversion; i. e., binary image input/output.

1 - Hollerith to display code for read; display code to Hollerith for punch.

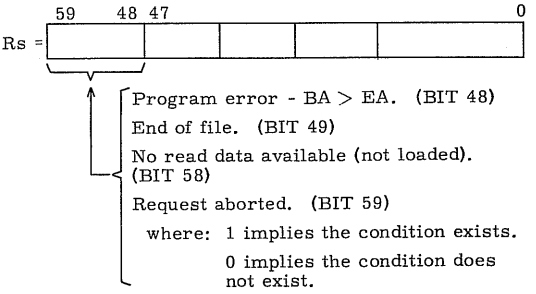
2 - Hollerith to BCD for read; BCD to Hollerith for punch.

STATUS RESPONSE CODES - positioned as per address S.

Rs = 0 Request is completed with no trouble.

Rs = 1 Request is in process.

Rs < 0 Request aborted. Reason given in bits 58 - 48.



5. CONSOLE OPERATIONS

Request procedures are provided for CP or ASPER routines to display messages on the primary console right scope or either of the scopes on other consoles. The system provides a timing service for removal of displays after a certain exposure. However, the request procedure gives an option to override the system time limit on display. In this mode, it is assumed that the CP or ASPER routine will request a removal of the display as a result of console acknowledgement or internal decision.

Opcode	Address Field	Remarks*
DSRW	N, S, BA, EA, RL, TAG, T	Display on Right Scope for system time limit.
DSLW	N, S, BA, EA, RL, TAG, T	Display on Left Scope for system time limit.
DHRW	N, S, BA, EA, RL, TAG, T	Display on Right Scope and hold indefinitely.
DHLW	N, S, BA, EA, RL, TAG, T	Display on Left Scope and hold indefinitely
RDPW	N, S, TAG	Remove display.
RTYW	N, S, BA, EA, RL, TAG	Read console typewriter.

*Wait if W is used.

N = Console logical unit number; 1, 2, . . . M for M consoles in the system.
S = Location containing central memory address for status response codes from System Periph. Processor I/O Routine.
BA = Location containing beginning address of buffer area in central memory.
EA = Location containing ending address + 1 of buffer area in central memory.
RL = Total number of characters in the message to be transmitted.
TAG = Identification number ≦ 18 bits for display message.
T = Display character size.

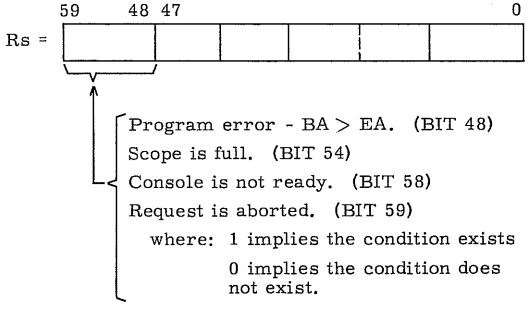
Blank or 0 - 64 characters/line.
1 - 32 characters/line.
2 - 16 characters/line.
3 - plot mode.

STATUS RESPONSE CODES - positioned as per address S.

Rs = 0 Request is completed with no trouble.

Rs = 1 Request is in process.

Rs < 0 Request aborted. Reason given in bits 58-48.



6. SYSTEM ACTION

Opcode	Address Field	Remarks*
TPPW	N, S, SYMBOL	Transfer program SYMBOL from CM to PP memory and begin execution with first ASPER instruction.
RQMW	NW, S, A	Request memory.
DRMW	NW, S, A	Release memory.
RQDW	N, S, L, NAME, R	Request disk space.
DRDW	N, S, NAME	Release disk space.
RQCW**	D, S	Request I/O channel.
DRCW**	D, S	Release I/O channel.
DRPP**	N, S	Release peripheral processor.

*Wait if W is used.

**These macros are used in ASPER program only.

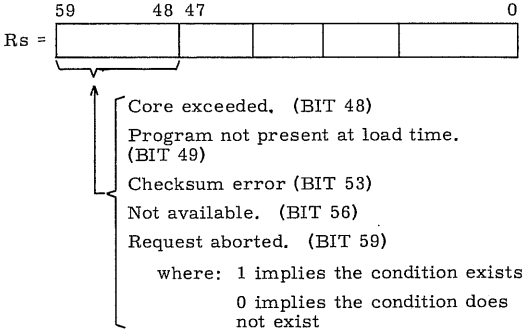
N = Logical number of PP or disk unit.
S = Location containing central memory address for status response code from System PP I/O routine.
D = Physical number of the I/O channel requested.
R = Maximum number of logical records into which the file may be segmented.
NW = Total number of words.
L = Number of 60-bit words in longest record.
A = Location containing central memory address of the first word of block assigned by the system or released by the programmer.
NAME = Symbolic name uniquely identifying the disk logical file being referenced.
SYMBOL = Name of PP program defined by ASPER pseudo operation.

STATUS RESPONSE CODES - positioned as per address S.

Rs = 0 Request completed with no trouble.

Rs = 1 Request in process.

Rs < 0 Request aborted. Reason given in bits 58 - 48.



7. CENTRAL PROCESSOR PROGRAM OVERLAY

During initial loading segmentation control, cards are matched against subroutines present to assure overlay capability when called. Therefore, control is taken from the central memory program during the load and is only returned when the load is successful. The loading could fail because of an attempt to load a non-existent segment or subroutine. No status is required since success is necessary to regain control.

Opcode	Address Field	Remarks
LOAD	SYMBOL	Load segment SYMBOL.
LOAD	*SYMBOL*	Load segment SYMBOL and transfer control to indicated routine.

SYMBOL = Name of overlay region to be loaded.

8. PERIPHERAL PROCESSOR PROGRAM OVERLAY

No execution takes place unless all SUBP's called in LOAD macros are present. During execution of the LOAD macro, control is kept in the macro and returned to the routine only upon successful completion of the load. Therefore, no status is provided.

Opcode	Address Field	Remarks
LOAD	SYMBOL	Load SUBP SYMBOL into periph. processor memory

SYMBOL = Name of overlay region to be loaded.

9. WAIT CHECK

After a buffered operation is initiated, a Wait Check macro may be used to check status. The routine delays until the status response word is zero (completed) or negative (aborted). If it is zero, the next instruction in line is executed. If the status word is negative, the routine exits to the location specified by SYMBOL.

Opcode	Address Field	Remarks
WAIW	S, SYMBOL	Check status of S. Exit to SYMBOL if abort. Wait for reply if not ready and W is used.

S = Location containing central memory address for status response code from System Periph. Processor I/O Routine.

SYMBOL = Transfer location if an abort is indicated by the status response code.

CONSOLE DISPLAY CODES

Char.	Code	Char.	Code
(space)	00	X	30
A	01	Y	31
B	02	Z	32
C	03	0	33
D	04	1	34
E	05	2	35
F	06	3	36
G	07	4	37
H	10	5	40
I	11	6	41
J	12	7	42
K	13	8	43
L	14	9	44
M	15	+	45
N	16	-	46
O	17	*	47
P	20	/	50
Q	21	(	51
R	22	)	52
S	23	blank	53
T	24	=	54
U	25	blank	55
V	26	,	56
W	27	.	57

Keyboard codes are identical with the following exceptions:

No Data	00
Carriage Return	60
Backspace	61
Space	62

PRINTER CODES (EXT. BCD)

CHAR.	CODE	CHAR.	CODE
(blank)	20	V	25
0	12	W	26
1	01	X	27
2	02	Y	30
3	03	Z	31
4	04	.	73
5	05	- (minus)	40
6	06	+	60
7	07	=	13
8	10	(	34
9	11	)	74
A	61	*	54
B	62	,	33
C	63	:	00
D	64	≠	14
E	65	/	21
F	66	≦	15
G	67	%	16
H	70	⊏	17
I	71	⊐	32
J	41	→	35
K	42	≡	36
L	43	∧ (and)	37
M	44	∨ (or)	52
N	45	$	53
O	46	↑	55
P	47	↓	56
Q	50	>	57
R	51	<	72
S	22	≧	75
T	23	¬ (not)	76
U	24	;	77

HOLLERITH PUNCH CARD CODES

CHAR.	CODE	CHAR.	CODE
A	12-1	Y	0-8
B	12-2	Z	0-9
C	12-3	0	0
D	12-4	1	1
E	12-5	2	2
F	12-6	3	3
G	12-7	4	4
H	12-8	5	5
I	12-9	6	6
J	11-1	7	7
K	11-2	8	8
L	11-3	9	9
M	11-4	/	0-1
N	11-5	+	12
O	11-6	- (dash)	11
P	11-7	blank	space
Q	11-8	.	12-8-3
R	11-9	)	12-8-4
S	0-2	$	11-8-3
T	0-3	*	11-8-4
U	0-4	,	0-8-3
V	0-5	(	0-8-4
W	0-6	=	8-3
X	0-7	_	8-4

CONTROL DATA
CORPORATION
COMPUTER DIVISION
4201 North Lexington Avenue St. Paul, Minnesot