Burroughs
B 5500
HANDBOOK

DISK/DATA COMM.
MCP
VERSION

GENERAL INFORMATION

PROCESSOR OPERATOR INDEX

I/O DESCRIPTOR INFORMATION

MESSAGES

DUMP DEBUGGING AIDS

MCP GENERAL INFORMATION

CHARACTER & WORD MODE OPERATIONS

INDEX OF DA'S AND PRINTS

MAINTENANCE INFORMATION

ROUTINES

1

Wherever There's Business There's **Burroughs**

Printed in U.S.A. America 10-67

1031986

Burroughs / **PUBLICATION CHANGE NOTICE**

PCN No.: 1031986-001 Date: August , 1970
Publication Title: Burroughs B 5500 Handbook

Other Affected Publications:

Supersedes:

Description

Replace the following Pages:		Add these Pages:
i/ii	6-1/6-2	4-89/4-90 thru 4-98
	6-3/6-4	
2-1/2-2	6-9/6-10	
2-7/2-8	6-11/6-12	
	6-13/6-14	
3-3/3-4	6-19/6-20	
3-9/3-10	6-21/6-22	
3-13/3-14		
3-19/3-20	7-3/7-4	
	7-15/7-16	
4-1 thru 4-88		
	8-1/8-2	
5-1/5-2		
5-3/5-4	9-9/9-10	
5-7/5-8		
5-11/5-12	10-11/10-12	
5-17/5-18	10-13/10-14	
5-19/5-20	10-15/10-16	
	10-17/10-18	

Printed in U.S. America

TABLE OF CONTENTS

SECTION	TITLE	PAGE
1	GENERAL INFORMATION	
	System Serial Numbers	1-1
	Troubleshooting Precautions	1-2
	Binary Card With Halt/Load Addresses	1-3
	Collating Sequence	1-4
	Data Transmission Codes	1-6
	Number Representation - Order of Magnitude Table	1-8
	Decimal and Octal Sample Conversion Problems	1-9
	Interrupts	1-13
2	PROCESSOR OPERATOR INDEX	
	Word Mode Operator Index - Numerical by Octal Code	2-1
	Word Mode Operator Index - Alphabetical by Operator	2-2
	Character Mode Operator Index - Numerical by Octal Code.	2-4
	Character Mode Operator Index - Alphabetical by Operator.	2-5
	Control State and Miscellaneous Operators Index - Numerical by Octal Code	2-6
	Control State and Miscellaneous Operators Index - Alphabetical by Operator	2-7
	Data and Control Words	2-7
3	I/O DESCRIPTOR INFORMATION	
	Composite Result Descriptor	3-1
	Magnetic Tape Mod. III I/O Result Descriptor in Memory.	3-2
	Disk File Result Descriptor	3-4
	Data Transmission Result Descriptor	3-5
	I/O Descriptors	3-6
	I/O Tape Descriptors	3-8
	Disk File Descriptors and Operation	3-11
	Disk File Lockout Chart	3-13
	Data Transmission Descriptor	3-14
	Standard Input Message from Answer * Back Drum on ASR Stations	3-15
	Data Transmission Registers	3-18
	Unit Designation	3-19
	Data Transmission Descriptor	3-20
4	MESSAGES	
	General	4-1
	System Messages	4-1
	Keyboard Output Messages	4-2
	Keyboard Input Messages	4-23
	The Remote SPO Station Facility	4-46
	Extended ALGOL Syntactical Error Messages	4-55
	COBOL Compiler Error and Diagnostic Messages	4-67
5	DUMP DEBUGGING AIDS	
	Dump Decoding Aids	5-1
	Descriptor Formats	5-1
	Cell Designation for Addresses One to One Hundred Octally.	5-2
	Array PRT [**]	5-3

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

TABLE OF CONTENTS - Continued

SECTION	TITLE	PAGE
5 - continued	Array Format	5-4
	Method for Declaring Array Space With DF MCP	5-5
	Interrogation Aids using an I/O Channel	5-5
	Use of Memory Load Switch in Troubleshooting	5-6
	Typical Stack Status at Time of Interrupt	5-6
	Memory Links	5-7
	Core Memory of Halt-Load Time Modules 0, 1, 3 and 4 on Line	5-8
	Segment Dictionary and Related PRT Cells as Created by the Compiler	5-9
	Array JAR [**] Jobs Actually Running	5-10
	Disk Communicates	5-12
	Option Word	5-13
	B 5500 Station Table Format	5-13
	Program and Dump Interrogation Worksheet	5-14
	Operand Call Syllable Flow Chart	5-16
	Descriptor Call Syllable Flow Chart	5-17
	Index Operations - Operand and Descriptor Call Syllable Flow Chart	5-18
	Subroutine Entry - Operand or Descriptor Call Syllable Flow Chart	5-19
6	MCP GENERAL INFORMATION	
	Main MCP PRT Locations (For MK VII DCMCP)	6-1
	Disk Layout	6-4
	Disk Directory	6-5
	DALOC-ARRAY DALOC [*] Size is 64 or @ 100	6-6
	DMIX-ARRAY DMIX [*] Size is MIXMAX+1	6-7
	Available - Disk Table	6-7
	SLATE [*]	6-7
	SHEET	6-8
	I/O-QUEUE	6-10
	Input Output Assignment Tables	6-11
	Array Information Table	6-12
	File Parameter Block (FPB) (Addressed by R+3)	6-13
	File Tank	6-14
	File Information Block (FIB)	6-15
	NFO	6-16
	Standard B 5500 Label Record	6-17
	Log Maintenance	6-17
	Printer Backup Information	6-26
	Closing a Print File on Disk	6-28
	Control Information	6-29
7	GENERAL INFORMATION	
	Operator Description	7-1
	Word Mode Operators	7-2
	Word Mode Operator - Subroutine Operators	7-13
	Character Mode Operators	7-15
	Special Character Mode Operators	7-22

TABLE OF CONTENTS - Continued

SECTION	TITLE	PAGE
8	INDEX OF DA'S AND PRINTS	
	Index of DA's and Prints	8-1
9	MAINTENANCE INFORMATION	
	Clock Test Points	9-1
	Processor Maintenance Panel Toggle Switches	9-1
	T Register Decoding - Word Mode	9-2
	B 460 Core Memory Over/Under Voltage Panel	9-10
	B 461 Core Memory Test Points	9-10
	B 461 Logical Gate - Wiring Side	9-11
	B 430 Drum Test Points	9-12
	Drum Card Rack - Wiring Side	9-12
10	TEST ROUTINES	
	Test Routines	10-1
	Test Routine Loader - TR5900	10-2

PRINTED IN U.S. AMERICA 10-67 1031986

SECTION 1

GENERAL INFORMATION

SYSTEM SERIAL NUMBERS

Unit Type	Name	Serial Number(s)
B5290	Display and Distribution	
B5281	Processor	
B5283	I/O Control	
B5220	Central Control	
B5370	Power Supply	
B5310	Control Console	
B5230	Drum Memory	
B5260	Core Memory	
B460/B461	Memory Module #0	
B460/B461	Memory Module #1	
B460/B461	Memory Module #2	
B460/B461	Memory Module #3	
B460/B461	Memory Module #4	
B460/B461	Memory Module #5, #6 & #7	
B122/B123/B124	Card Reader #1	
B122/B123/B124	Card Reader #2	
B320/B321	Line Printer #1	
B320/B321	Line Printer #2	
B303/B304	Card Punch	
B141	Paper Tape Reader #1	
B141	Paper Tape Reader #2	
B341	Paper Tape Punch #1	
B341	Paper Tape Punch #2	
B422/B423	Magnetic Tape Unit A	
B422/B423	Magnetic Tape Unit B	
B422/B423	Magnetic Tape Unit C	
B422/B423	Magnetic Tape Unit D	

PRINTED IN U.S. AMERICA 10-67 1031986

TROUBLESHOOTING PRECAUTIONS

1. Do not use a battery-buzzer for continuity checking. The buzzer current exceeds the maximum current rating for diodes and transistors in the system.

2. Do not use the first two low scales (X1 or X10) on the Triplett ohmmeter for continuity checking. For these scales, the meter current exceeds the maximum current rating for diodes and transistors in the system.

3. Do not remove packages or diode sticks when DC Power is ON. Do not remove Inhibit Drivers or Driver Switch Packages with AC Power ON.

4. Care must be taken when using Scope or Jumper Clip Leads to prevent touching adjacent pins. Use Minigator Clips with insulators or the Wire Wrap Pin Probe Tip (Part No. 11838547).

5. Use extreme caution when working on the plug-in side of the panels. Avoid hitting packages when moving the scope.

6. Do not attempt to force a TRUE level with -12V. In all cases, the desired effect can be obtained either by the use of a ground clip, or by taping off one or more diode contacts.

7. A ground jumper may be used to force a FALSE level.

NOTE

Connect clip to the point to be grounded prior to making ground connection.

8. Do not pull Cable Plugs with POWER ON at either end of the cable.

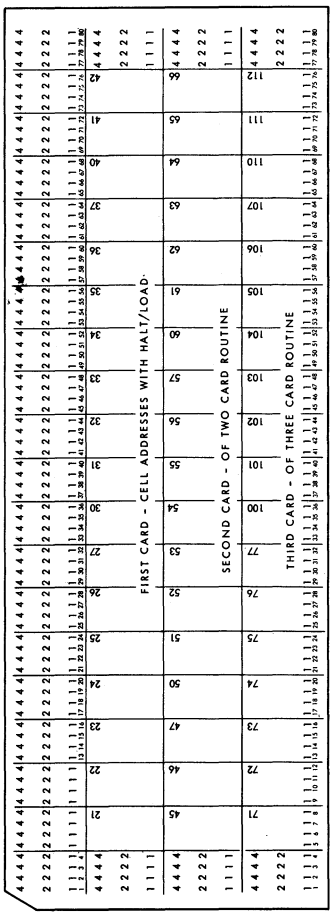
9. Only soldering irons that have an isolation transformer may be used.

10. Scope ground - to prevent ground loops and noise interference use only the ground clip on the scope probe. Attach it to a suitable ground as near as possible to the point being observed.

11. If a separate ground for the scope is used, make sure the ground lead is on logic DC ground and not frame ground.

12. -24V can be removed only at remote circuit breaker.

BINARY CARD WITH HALT/LOAD ADDRESSES



PRINTED IN U.S. AMERICA 10-67 1031986

COLLATING SEQUENCE

CHAR.	INTERNAL CODE			BCL CODE		CARD CODE	
	BA	8421	OCTAL CODE	BA	8421	ZONE	NUM.
Blank	11	0000	60	01	0000	-	-
.	01	1010	32	11	1011	12	8 - 3
[	01	1011	33	11	1100	12	8 - 4
(	01	1101	35	11	1101	12	8 - 5
<	01	1110	36	11	1110	12	8 - 6
←	01	1111	37	11	1111	12	8 - 7
&	01	1100	34	11	0000	12	-
$	10	1010	52	10	1011	11	8 - 3
*	10	1011	53	10	1100	11	8 - 4
)	10	1101	55	10	1101	11	8 - 5
;	10	1110	56	10	1110	11	8 - 6
≤	10	1111	57	10	1111	11	8 - 7
-	10	1100	54	10	0000	11	-
/	11	0001	61	01	0001	0	1
,	11	1010	72	01	1011	0	8 - 3
%	11	1011	73	01	1100	0	8 - 4
=	11	1101	75	01	1101	0	8 - 5
]	11	1110	76	01	1110	0	8 - 6
"	11	1111	77	01	1111	0	8 - 7
#	00	1010	12	00	1011	-	8 - 3
@	00	1011	13	00	1100	-	8 - 4
:	00	1101	15	00	1101	-	8 - 5
>	00	1110	16	00	1110	-	8 - 6
≥	00	1111	17	00	1111	-	8 - 7
+	01	0000	20	11	1010	12	0
A	01	0001	21	11	0001	12	1
B	01	0010	22	11	0010	12	2
C	01	0011	23	11	0011	12	3
D	01	0100	24	11	0100	12	4
E	01	0101	25	11	0101	12	5
F	01	0110	26	11	0110	12	6
G	01	0111	27	11	0111	12	7

LOW → COLLATING SEQUENCE → HIGH

COLLATING SEQUENCE (continued)

CHAR.	INTERNAL CODE			BCL CODE		CARD CODE	
	BA	8421	OCTAL CODE	BA	8421	ZONE	NUM.
H	01	1000	30	11	1000	12	8
I	01	1001	31	11	1001	12	9
×	10	0000	40	10	1010	11	0
J	10	0001	41	10	0001	11	1
K	10	0010	42	10	0010	11	2
L	10	0011	43	10	0011	11	3
M	10	0100	44	10	0100	11	4
N	10	0101	45	10	0101	11	5
O	10	0110	46	10	0110	11	6
P	10	0111	47	10	0111	11	7
Q	10	1000	50	10	1000	11	8
R	10	1001	51	10	1001	11	9
≠	11	1100	74	01	1010	0	8 - 2
S	11	0010	62	01	0010	0	2
T	11	0011	63	01	0011	0	3
U	11	0100	64	01	0100	0	4
V	11	0101	65	01	0101	0	5
W	11	0110	66	01	0110	0	6
X	11	0111	67	01	0111	0	7
Y	11	1000	70	01	1000	0	8
Z	11	1001	71	01	1001	0	9
0	00	0000	00	00	1010	-	0
1	00	0001	01	00	0001	-	1
2	00	0010	02	00	0010	-	2
3	00	0011	03	00	0011	-	3
4	00	0100	04	00	0100	-	4
5	00	0101	05	00	0101	-	5
6	00	0110	06	00	0110	-	6
7	00	0111	07	00	0111	-	7
8	00	1000	10	00	1000	-	8
9	00	1001	11	00	1001	-	9
?	00	1100	14	00	0000	ALL OTHER CARD CODES	

LOW → COLLATING SEQUENCE → HIGH

PRINTED IN U.S. AMERICA 10-67

1031986

DATA TRANSMISSION CODES

BCL	ASCII	BAUDOT	BCL		TYP/TWX ASCII		TTY BAUDOT		
			BA	8421	765	4321	SHIFT	5	4321
BLANK	SPACE	SPACE	01	0000	010	0000	0	0	0100
	:		11	1011	010	1110	1	1	1100
[	[	FIG SFT	11	1100	101	1011	1	1	1011
(	(	(	11	1101	010	1000	1	0	1111
<	<		11	1110	011	1100			
	RUB OUT				111	1111			
←	←	BLANK	11	1111	101	1111	0	0	0000
&	&	&	11	0000	010	0110	1	1	1010
$	$	$	10	1011	010	0100	1	0	1001
*	*	'	10	1100	010	1010	1	0	1011
)	)	)	10	1101	010	1001	1	1	0010
;	;	;	10	1110	011	1011	1	1	1110
≤	'	CR	10	1111	010	0111	0	0	1000
	CAR RET				000	1101			
-	-	-	10	0000	010	1101	1	0	0011
/	/	/	01	0001	010	1111	1	1	1101
,	,	,	01	1011	010	1100	1	0	1100
%	%		01	1100	010	0101			
=	=		01	1101	011	1101			
]	]	LTR SFT	01	1110	101	1101	0	1	1111
"	"	"	01	1111	010	0010	1	1	0001
#	#	(UPPER H) DISCONN	00	1011	010	0011	1	1	0100
@	@		00	1100	100	0000			
:	:	:	00	1101	011	1010	1	0	1110
>	>		00	1110	011	1110			
	X-ON				001	0001			
≥	!	!	00	1111	010	0001	1	0	1101
	DISCONN								
+	+		11	1010	010	1011			
A	A	A	11	0001	100	0001	0	0	0011
B	B	B	11	0010	100	0010	0	1	1001
C	C	C	11	0011	100	0011	0	0	1110
D	D	D	11	0100	100	0100	0	0	1001
E	E	E	11	0101	100	0101	0	0	0001
F	F	F	11	0110	100	0110	0	0	1101
G	G	G	11	0111	100	0111	0	1	1010
H	H	H	11	1000	100	1000	0	1	0100
I	I	I	11	1001	100	1001	0	0	0110
x	\	BELL	10	1010	101	1100	1	0	0101
	FORM OUT				000	1100			
J	J	J	10	0001	100	1010	0	0	1011
K	K	K	10	0010	100	1011	0	0	1111
L	L	L	10	0011	100	1100	0	1	0010
M	M	M	10	0100	100	1101	0	1	1100
N	N	N	10	0101	100	1110	0	0	1100
O	O	O	10	0110	100	1111	0	1	1000
P	P	P	10	0111	101	0000	0	1	0110
Q	Q	Q	10	1000	101	0001	0	1	0111

DATA TRANSMISSION CODES (continued)

BCL	ASCII	BAUDOT	BCL		TYP/TWX ASCII		TTY BAUDOT		
			BA	8421	765	4321	SHIFT	5	4321
R	R	R	10	1001	101	0010	0	0	1010
≠	↓ LIN FEED	LIN FEED	01	1010	101	1110	0	0	0010
					000	1010			
S	S	S	01	0010	101	0011	0	0	0101
T	T	T	01	0011	101	0100	0	1	0000
U	U	U	01	0100	101	0101	0	0	0111
V	V	V	01	0101	101	0110	0	1	1110
W	W	W	01	0110	101	0111	0	1	0011
X	X	X	01	0111	101	1000	0	1	1101
Y	Y	Y	01	1000	101	1001	0	1	0101
Z	Z	Z	01	1001	101	1010	0	1	0001
0	0	0	00	1010	011	0000	1	1	0110
1	1	1	00	0001	011	0001	1	1	0111
2	2	2	00	0010	011	0010	1	1	0011
3	3	3	00	0011	011	0011	1	0	0001
4	4	4	00	0100	011	0100	1	0	1010
5	5	5	00	0101	011	0101	1	1	0000
6	6	6	00	0110	011	0110	1	1	0101
7	7	7	00	0111	011	0111	1	0	0111
8	8	8	00	1000	011	1000	1	0	0110
9	9	9	00	1001	011	1001	1	1	1000
?		?	00	0000	011	1111	1	1	1001

NOTE: BCL CODE HAS EVEN PARITY — (EVEN NUMBER OF 1 BITS).

PRINTED IN U.S. AMERICA 10-67

1031986

OCTAL MULTIPLY CHART

1	1						
2	2	4					
3	3	6	11				
4	4	10	14	20			
5	5	12	17	24	31		
6	6	14	22	30	36	44	
7	7	16	25	34	43	52	61
	1	2	3	4	5	6	7

NUMBER REPRESENTATION - ORDER OF MAGNITUDE TABLE

REGISTER BIT SET	NUMERIC EQUIVALENT	OCTAL	BINARY
1	1 1.0	8^0	2^0
2	2 0.5		
3	4 0.25		
4	8 0.125	8^1	2^3
5	16 0.062 5		
6	32 0.031 25		
7	64 0.015 625	8^2	2^6
8	128 0.007 812 5		
9	256 0.003 906 25		
10	512 0.001 953 125	8^3	2^9
11	1 024 0.000 976 562 5		
12	2 048 0.000 488 281 25		
13	4 096 0.000 244 140 625	8^4	2^{12}
14	8 192 0.000 122 070 312 5		
15	16 384 0.000 061 035 156 25		
16	32 768 0.000 030 517 578 125	8^5	2^{15}
17	65 536 0.000 015 258 789 062 5		
18	131 072 0.000 007 629 394 531 25		
19	262 144 0.000 003 814 697 265 625	8^6	2^{18}
20	524 288		
21	1 048 576		
22	2 097 152	8^7	2^{21}
23	4 194 304		
24	8 388 608		
25	16 777 216	8^8	2^{24}
26	33 554 432		
27	67 108 864		
28	134 217 728	8^9	2^{27}
29	268 435 456		
30	536 870 912		
31	1 073 741 824	8^{10}	2^{30}
32	2 147 483 648		
33	4 294 967 296		
34	8 589 934 592	8^{11}	2^{33}
35	17 179 869 184		
36	34 359 738 368		
37	68 719 476 736	8^{12}	2^{36}
38	137 438 953 472		
39	274 877 906 944		
ALL 39	549 755 813 887	8^{13}	2^{39}
	549 755 813 888		

99999999_D $575\ 360\ 377_O$

DECIMAL AND OCTAL SAMPLE CONVERSION PROBLEMS

DECIMAL 531 = OCTAL 1023

66 - remainder 3
8/531

8 - remainder 2
8/66

1 - remainder 0
8/8

0 - remainder 1
8/1

1 0 2 3 ans.

DECIMAL .439453125 = OCTAL .341

.439453125
× 8
3.515625000
.515625
× 8
4.125000
.125
× 8
1.000

ans. .3 4 1

OCTAL 2672 = DECIMAL 1466

2 6 7 2
× 8
16 + 6 = 22
× 8
176 + 7 = 183
× 8
1464 + 2 = 1466

OCTAL .341 = DECIMAL .439453125

.3 4 1

.125
8/1.000
.515625
8/4.125000
.439453125 ans.
8/3.515625000

PRINTED IN U.S. AMERICA 10-67

1031906

TABLE 9-10. OCTAL: DECIMAL CONVERSION.

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383

TABLE 9-10. OCTAL: DECIMAL CONVERSION (continued)

	0	1	2	3	4	5	6	7
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

PRINTED IN U.S. AMERICA 10-67

1031986

TABLE 9-10. OCTAL: DECIMAL CONVERSION (continued)

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

INTERRUPTS

PRIORITY

TYPE	INDICATION	ØCTAL CELL
P1 MEMORY PARITY ERROR	Pk-101F	60
P1 INVALID ADDRESS	Pk-102F	61
TIME INTERVAL	CC 103F	22
I/O BUSY	CC 104F	23
KEYBOARD REQUEST	CC 105F	24
I/O #1 FINISHED	CC 108F	27
I/O #2 FINISHED	CC 109F	30
I/O #3 FINISHED	CC 110F	31
I/O #4 FINISHED	CC 111F	32
PRINTER 1 FINISHED	CC 106F	25
PRINTED 2 FINISHED	CC 107F	26
P2 BUSY	CC 112F	33
INQUIRY REQUEST	CC 113F	34
SPECIAL INTERRUPT 1	CC 114F	35
DF READ CHECK FINISHED 1	CC 115F	36
DF READ CHECK FINISHED 2	CC 116F	37
* P1 STACK OVERFLOW	Pk-103F	62
P1 COMMUNICATE	Pk BCD 4	64
P1 PROGRAM RELEASE	Pk BCD 5	65
P1 CONTINUITY BIT	Pk BCD 6	66
P1 PRESENCE BIT	Pk BCD 7	67
P1 FLAG BIT	Pk BCD 8	70
P1 INVALID INDEX	Pk BCD 9	71
P1 EXPONENT UNDERFLOW	Pk BCD 10	72
P1 EXPONENT OVERFLOW	Pk BCD 11	73
P1 INTEGER OVERFLOW	Pk BCD 12	74
P1 DIVIDE BY ZERO	Pk BCD 13	75
P2 MEMORY PARITY ERROR	Pk-101F	40
P2 INVALID ADDRESS	Pk-102F	41
P2 STACK OVERFLOW	Pk-103F	42
P2 COMMUNICATE	Pk BCD 4	44
P2 PROGRAM RELEASE	Pk BCD 5	45
P2 CONTINUITY BIT	Pk BCD 6	46
P2 PRESENCE BIT	Pk BCD 7	47
P2 FLAG BIT	Pk BCD 8	50
P2 INVALID INDEX	Pk BCD 9	51
P2 EXPONENT UNDERFLOW	Pk BCD 10	52
P2 EXPONENT OVERFLOW	Pk BCD 11	53
P2 INTEGER OVERFLOW	Pk BCD 12	54
P2 DIVIDE BY ZERO	Pk BCD 13	55

* A PSEUDO STACK OVERFLOW IS CREATED WHEN THE MCP ATTEMPTS TO INITIATE A PROGRAM WHOSE R+0 DOES NOT CONTAIN @2525252525252525

PRINTED IN U.S. AMERICA 10-67

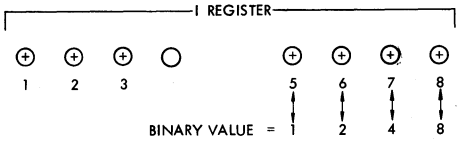
1031986

INTERRUPTS (continued)

CENTRAL CONTROL

INDICATION	TYPE	ØCTAL CELL
CC 103F	TIME INTERVAL	22
CC 104F	I/O BUSY	23
CC 105F	KEYBOARD REQUEST	24
CC 108F	I/O #1 FINISHED	27
CC 109F	I/O #2 FINISHED	30
CC 110F	I/O #3 FINISHED	31
CC 111F	I/O #4 FINISHED	32
CC 106F	PRINTER 1 FINISHED	25
CC 107F	PRINTER 2 FINISHED	26
CC 112F	P2 BUSY	33
CC 113F	INQUIRY REQUEST	34
CC 114F	NOT ASSIGNED	35
CC 115F	DISK FILE #1 FINISHED	36
CC 116F	DISK FILE #2 FINISHED	37

PROCESSOR



INDICATION	ØCTAL CELL		TYPE
	P1	P2	
Pk-101F	60	40	MEMORY PARITY ERROR
Pk-102F	61	41	INVALID ADDRESS
Pk-103F	62	42	STACK OVERFLOW
Pk BCD 4	64	44	COMMUNICATE
Pk BCD 5	65	45	PROGRAM RELEASE
Pk BCD 6	66	46	CONTINUITY BIT
Pk BCD 7	67	47	PRESENCE BIT
Pk BCD 8	70	50	FLAG BIT
Pk BCD 9	71	51	INVALID INDEX
Pk BCD 10	72	52	EXPONENT UNDERFLOW
Pk BCD 11	73	53	EXPONENT OVERFLOW
Pk BCD 12	74	54	INTEGER OVERFLOW
Pk BCD 13	75	55	DIVIDE BY ZERO

SECTION 2
PROCESSOR OPERATOR INDEX

WORD MODE OPERATOR INDEX - NUMERICAL BY OCTAL CODE

OCTAL CODE	MNEMONIC ENG.	MNEMONIC ESPOL	FLOW CHART	OPERATOR
LS45	VFIL	ISO	1.34.0	VARIABLE FIELD ISOLATE
XX55	DIAL	DIA	1.19.0	DIAL A
XX61	DIBL	DIB	1.19.0	DIAL B
XX65	TRFL	TRB	1.20.0	TRANSFER BITS
XX71	CFLL	FCL	1.22.0	COMPARE FIELD LOW
XX75	CFEL	FCE	1.21.0	COMPARE FIELD EQUAL
X051	ZFNL	CFN	1.33.0	BRANCH FORWARD NONDESTRUCTIVE
X151	ZBNL	CBN	1.33.0	BRANCH BACKWARD NONDESTRUCTIVE
X251	ZFDL	CFD	1.33.0	BRANCH FORWARD DESTRUCTIVE
X351	ZBDL	CBD	1.33.0	BRANCH BACKWARD DESTRUCTIVE
X451	ZFNL	CFN	1.33.0	BRANCH FORWARD NONDESTRUCTIVE
X551	ZBNL	CBN	1.33.0	BRANCH BACKWARD NONDESTRUCTIVE
X651	ZFDL	CFD	1.33.0	BRANCH FORWARD DESTRUCTIVE
X751	ZBDL	CBD	1.33.0	BRANCH BACKWARD DESTRUCTIVE
0051	DELL	DEL	1.33.0	DELETE
0101	AD1L	ADD	1.01.0	SINGLE PRECISION ADD
0105	AD2L	DLA	1.02.0	DOUBLE PRECISION ADD
0115	LUNL	LNG	1.12.0	LOGICAL "NEGATE"
0121	CSDL	CID	1.18.0	COND. INTEGER STORE DEST.
0125	BGEL	GEQ	1.13.0	B GREATER OR EQUAL TO A
0131	BBCL	BBC	1.15.0	BRANCH BACKWARD CONDITIONAL
0135	RJPL	BRT	1.16.0	BRANCH RETURN
0141	INDL	INX	1.31.0	INDEX
0215	LOOL	LOR	1.10.0	LOGICAL "OR"
0221	CSNL	CIN	1.18.0	COND. INTEGER STORE NONDEST.
0225	BGAL	GTR	1.13.0	B GREATER THAN A
0231	BFCL	BFC	1.15.0	BRANCH FORWARD CONDITIONAL
0235	RNML	RTN	1.27.0	RETURN NORMAL
0241	MDVL	COC	1.32.0	CONSTRUCT OPERAND CALL
0301	SU1L	SUB	1.01.0	SINGLE PRECISION SUBTRACT
0305	SU2L	DLS	1.02.0	DOUBLE PRECISION SUBTRACT
0401	MU1L	MUL	1.03.0	SINGLE PRECISION MULTIPLY
0405	MU2L	DLM	1.04.0	DOUBLE PRECISION MULTIPLY
0415	LOAL	LND	1.09.0	LOGICAL "AND"
0421	BSDL	STD	1.17.0	B STORE DESTRUCTIVE
0425	BNEL	NEQ	1.13.0	B NOT EQUAL TO A
0431	MSNL	SSN	1.25.0	SET SIGN BIT
0441	MSOL	MKS	1.26.0	MARK STACK
1001	DV1L	DIV	1.05.0	SINGLE PRECISION DIVIDE
1005	DV2L	DLD	1.06.0	DOUBLE PRECISION DIVIDE
1015	LOEL	LQV	1.11.0	LOGICAL "EQUIVALENCE"
1021	BSNL	SND	1.17.0	B STORE NONDESTRUCTIVE
1025	EXCL	XCH	1.28.0	EXCHANGE
1031	CSSL	CHS	1.25.0	CHANGE SIGN BIT
1235	RSPL	RTS	1.27.0	RETURN SPECIAL
1241	MDAL	CDC	1.32.0	CONSTRUCT DESCRIPTOR CALL
1425	FCXL	FTC	1.28.0	F FIELD TO CORE FIELD

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

WORD MODE OPERATOR INDEX - NUMERICAL BY OCTAL CODE (continued)

OCTAL CODE	MNEMONIC ENG.	MNEMONIC ESPOL	FLOW CHART	OPERATOR
2015	FBL	MOP	1.23.0	RESET FLAG BIT
2021	CODL	LOD	1.30.0	LOAD
2025	DUPL	DUP	1.29.0	DUPLICATE
2031	TFBL	TOP	1.24.0	TEST FLAG BIT
2131	JBCL	LBC	1.14.0	BRANCH BKWD. WORD CONDITIONAL
2141	FXSL	SSF	1.31.0	F & S REG. SET/STORE
2141	FXSL	SSF	1.31.0	A REG. = 0 STORE F
2141	FXSL	SSF	1.31.0	A REG. = 1 STORE S
2141	FXSL	STF	1.31.0	A REG. = 2 SET F
2141	FXSL	STS	1.31.0	A REG. = 3 SET S
2231	JFCL	LFC	1.14.0	BRANCH FWD. WORD CONDITIONAL
2431	IPSL	TUS	1.24.0	INTERROGATE PERIPHERAL STATUS
2541	LLLL	LLL	1.31.0	LINK LIST LOOK UP
3001	DU3L	IDV	1.07.0	INTEGER DIVIDE
3425	FFXL	FTF	1.28.0	F FIELD TO F FIELD
4015	SFBL	MDS	1.23.0	SET FLAG BIT
4121	ISDL	ISD	1.18.0	INTEGER STORE DESTRUCTIVE
4125	BLEL	LEQ	1.13.0	B LESS THAN OR EQUAL TO A
4131	BBUL	BBW	1.14.0	BRANCH BACKWARD UNCONDITIONAL
4221	ISNL	ISN	1.18.0	INTEGER STORE NONDESTRUCTIVE
4225	BLAL	LSS	1.13.0	B LESS THAN A
4231	BFUL	BFW	1.14.0	BRANCH FORWARD UNCONDITIONAL
4425	BEQL	EQL	1.13.0	B EQUAL TO A
4431	MSPL	SSP	1.25.0	RESET SIGN BIT
4441	ECML	CMN	1.26.0	ENTER CHARACTER MODE
5425	CCXL	CTC	1.28.0	CORE FIELD TO C FIELD
6131	JBUL	LBU	1.15.0	BRANCH BKWD. WORD UNCOND.
6231	JFUL	LFU	1.15.0	BRANCH FWD. WORD UNCOND.
6431	TIOL	TIO	1.24.0	INTERROGATE I/O CHANNEL
7001	DU4L	RDV	1.08.0	REMAINDER DIVIDE
7031	SSFL	FBS	1.25.0	STACK SEARCH FOR FLAG
7425	CFXL	CTF	1.28.0	CORE FIELD TO F FIELD

WORD MODE OPERATOR INDEX - ALPHABETICAL BY OPERATOR

OPERATOR	MNEMONIC ENG.	MNEMONIC ESPOL	OCTAL CODE	FLOW CHART
B EQUAL TO A	BEQL	EQL	4425	1.13.0
B GREATER OR EQUAL TO A	BGEL	GEQ	0125	1.13.0
B GREATER THAN A	BGAL	GTR	0225	1.13.0
B LESS OR EQUAL TO A	BLEL	LEQ	4125	1.13.0
B LESS THAN A	BLAL	LSS	4225	1.13.0
B NOT EQUAL TO A	BNEL	NEQ	0425	1.13.0
B STORE DESTRUCTIVE	BSDL	STD	0421	1.17.0
B STORE NONDESTRUCTIVE	BSNL	SND	1021	1.17.0
BRANCH BKWD. SYLL. CONDITIONAL	BBCL	BBC	0131	1.14.0
BRANCH BKWD. SYLL. UNCONDITIONAL	BBUL	BBW	4131	1.15.0
BRANCH BKWD. WORD CONDITIONAL	JBCL	LBC	2131	1.14.0
BRANCH BKWD. WORD UNCONDITIONAL	JBUL	LBU	6131	1.15.0
BRANCH BKWD. NONZERO DESTRUCTIVE	ZBDL	CBD	X351/X751	1.33.0
BRANCH BKWD. NONZERO NONDEST.	ZBNL	CBN	X151/X551	1.33.0
BRANCH FWD. SYLL. CONDITIONAL	BFCL	BFC	0231	1.14.0
BRANCH FWD. SYLL. UNCONDITIONAL	BFUL	BFW	4231	1.15.0
BRANCH FWD. WORD CONDITIONAL	JFCL	LFC	2231	1.14.0

WORD MODE OPERATOR INDEX - ALPHABETICAL BY OPERATOR (continued)

OPERATOR	MNEMONIC ENG.	MNEMONIC ESPOL	OCTAL CODE	FLOW CHART
BRANCH FWD. WORD UNCONDITIONAL	JFUL	LFU	6231	1.15.0
BRANCH FWD. NONZERO DESTRUCTIVE	ZFDL	CFD	X251/X651	1.33.0
BRANCH FWD. NONZERO NONDEST.	ZFNL	CFN	X051/X451	1.33.0
BRANCH RETURN	RJPL	BRT	0135	1.16.0
CHANGE SIGN BIT	CSSL	CHS	1031	1.25.0
COMPARE FIELD EQUAL	CFEL	FCE	XX75	1.21.0
COMPARE FIELD LOW	CFLL	FCL	XX71	1.22.0
COND. INTEGER STORE DEST.	CSDL	CID	0121	1.18.0
COND. INTEGER STORE NONDEST.	CSNL	CIN	0221	1.18.0
CONSTRUCT DESCRIPTOR CALL	MDAL	CDC	1241	1.32.0
CONSTRUCT OPERAND CALL	MDUL	COC	0241	1.32.0
CORE FIELD TO C FIELD	CCXL	CTC	5425	1.28.0
CORE FIELD TO F FIELD	CFXL	CTF	7425	1.28.0
DELETE	DELL	DEL	0051	1.33.0
DIAL A	DIAL	DIA	XX55	1.19.0
DIAL B	DIBL	DIB	XX61	1.19.0
DOUBLE PRECISION ADD	AD2L	DLA	0105	1.02.0
DOUBLE PRECISION DIVIDE	DU2L	DLD	1005	1.06.0
DOUBLE PRECISION MULTIPLY	MU2L	DLM	0405	1.04.0
DOUBLE PRECISION SUBTRACT	SU2L	DLS	0305	1.02.0
DUPLICATE	DUPL	DUP	2025	1.29.0
ENTER CHARACTER MODE	ECML	CMN	4441	1.26.0
EXCHANGE	EXCL	XCH	1025	1.28.0
F FIELD TO CORE FIELD	FCXL	FTC	1425	1.28.0
F FIELD TO F FIELD	FFXL	FTF	3425	1.28.0
F & S REG. SET/STORE	FXSL	SSF	2141	1.31.0
STORE F - A REG. = 0	FXSL	SSF	2141	1.31.0
STORE S - A REG. = 1	FXSL	SSF	2141	1.31.0
SET F - A REG. = 2	FXSL	SSF	2141	1.31.0
SET S - A REG. = 3	FXSL	SSF	2141	1.31.0
INDEX	INDL	INX	0141	1.31.0
INTEGER DIVIDE	DU3L	IDV	3001	1.07.0
INTERROGATE I/O CHANNEL	TIOL	TIO	6431	1.24.0
INTERROGATE PERIPHERAL STATUS	IPSL	TUS	2431	1.24.0
INTEGER STORE DESTRUCTIVE	ISDL	ISD	4121	1.18.0
INTEGER STORE NONDESTRUCTIVE	ISNL	ISN	4221	1.18.0
LINK LIST LOOK UP	LLLL	LLL	2541	1.31.0
LOAD	LODL	LOD	2021	1.30.0
LOGICAL "AND"	LOAL	LND	0415	1.09.0
LOGICAL "EQUIVALENCE"	LOEL	LQU	1015	1.11.0
LOGICAL "NEGATE"	LUNL	LNG	0115	1.12.0
LOGICAL "OR"	LOOL	LOR	0215	1.10.0
MARK STACK	MSOL	MKS	0441	1.26.0
REMAINDER DIVIDE	DV4L	RDV	7001	1.08.0
RESET FLAG BIT	FBL	MOP	2015	1.23.0
RESET SIGN BIT	MSPL	SSP	4431	1.25.0
RETURN NORMAL	RNML	RTN	0235	1.27.0
RETURN SPECIAL	RSPL	RTS	1235	1.27.0
SET FLAG BIT	SFBL	MDS	4015	1.23.0
SET SIGN BIT	MSNL	SSN	0431	1.25.0
SINGLE PRECISION ADD	AD1L	ADD	0101	1.01.0
SINGLE PRECISION DIVIDE	DV1L	DIV	1001	1.05.0

PRINTED IN U.S. AMERICA 10-67

1031986

WORD MODE OPERATOR INDEX - ALPHABETICAL BY OPERATOR (continued)

OPERATOR	MNEMONIC ENG.	MNEMONIC ESPOL	OCTAL CODE	FLOW CHART
SINGLE PRECISION MULTIPLY	MUTL	MUL	0401	1.03.0
SINGLE PRECISION SUBTRACT	SUTL	SUB	0301	1.01.0
STACK SEARCH FOR FLAG	SSFL	FBS	7031	1.25.0
TEST FLAG BIT	TFBL	TOP	2031	1.24.0
TRANSFER BITS	TRFL	TRB	XX65	1.20.0
VARIABLE FIELD ISOLATE	VFIL	ISO	LS45	1.34.0

CHARACTER MODE OPERATOR INDEX - NUMERICAL BY OCTAL CODE

OCTAL CODE	MNEMONIC ENG.	MNEMONIC ESPOL	FLOW CHART	OPERATOR
XX00	RECL	EXC	2.39.0	EXIT CHARACTER MODE
XX02	SBDL	BSD	2.32.0	SKIP BIT DESTINATION
XX03	SBSL	BSS	2.31.0	SKIP BIT SOURCE
XX04	RDAL	RDA	2.16.0	RECALL DESTINATION ADDRESS
XX05	TWDL	TRW	2.27.0	TRANSFER WORDS
XX06	SDPL	SED	2.14.0	SET DESTINATION ADDRESS
XX07	SDAL	TDA	2.13.0	TRANSFER DESTINATION ADDRESS
XX12	TBZL	TBN	2.05.0	TRANSFER BLANK FOR NONNUMERIC
XX14	STDL	SDA	2.15.0	STORE DESTINATION ADDRESS
XX15	STSL	SSA	2.11.0	STORE SOURCE ADDRESS
XX16	FSDL	SFD	2.25.0	SKIP FORWARD DESTINATION
XX17	RSDL	SRD	2.26.0	SKIP REVERSE DESTINATION
XX22	SSPL	SES	2.10.0	SET SOURCE ADDRESS
XX24	TEQL	TEQ	2.07.0	TEST FOR EQUAL
XX25	TNEL	TNE	2.07.0	TEST FOR NOT EQUAL
XX26	TGEL	TEG	2.07.0	TEST FOR GREATER OR EQUAL
XX27	TGTL	TGR	2.07.0	TEST FOR GREATER
XX30	RSSL	SRS	2.24.0	SKIP REVERSE SOURCE
XX31	FSSL	SFS	2.24.0	SKIP FORWARD SOURCE
XX32	FSXL	—	2.28.0	FIELD SUBTRACT (AUX)
XX33	FAXL	—	2.28.0	FIELD ADD (AUX)
XX34	TLEL	TEL	2.07.0	TEST FOR EQUAL OR LESS
XX35	TLTL	TLS	2.07.0	TEST FOR LESS
XX36	TANL	TAN	2.08.0	TEST FOR ALPHANUMERIC
XX37	TEBL	BIT	2.30.0	TEST BIT
XX40	INTL	INC	2.35.0	INCREASE TALLY
XX41	STAL	STC	2.34.0	STORE TALLY
XX42	SETL	SEC	2.35.0	SET TALLY
XX43	CLRL	CRF	2.33.0	CALL REPEAT FIELD
XX44	CJOL	JNC	2.21.0	JUMP OUT OF LOOP CONDITIONAL
XX45	CFJL	JFC	2.22.0	JUMP FORWARD CONDITIONAL
XX46	JOLL	JNS	2.21.0	JUMP OUT OF LOOP
XX47	FWJL	JFW	2.22.0	JUMP FORWARD UNCONDITIONAL
XX50	RPAL	RCA	2.18.0	RECALL CONTROL ADDRESS
XX51	ENLL	ENS	2.20.0	END LOOP
XX52	BELL	BNS	2.19.0	BEGIN LOOP
XX53	RSAL	RSA	2.12.0	RECALL SOURCE ADDRESS
XX54	STPL	SCA	2.17.0	STORE CONTROL ADDRESS
XX55	CRJL	JRC	2.23.0	JUMP REVERSE CONDITIONAL
XX56	SSAL	TSA	2.09.0	TRANSFER SOURCE ADDRESS
XX57	REJL	JRV	2.23.0	JUMP REVERSE UNCONDITIONAL

CHARACTER MODE OPERATOR INDEX - NUMERICAL BY OCTAL CODE (continued)

OCTAL CODE	MNEMONIC ENG.	MNEMONIC ESPOL	FLOW CHART	OPERATOR
XX60	SEQL	CEQ	2.06.0	COMPARE EQUAL
XX61	SNEL	CNE	2.06.0	COMPARE NOT EQUAL
XX62	SGEL	CEG	2.06.0	COMPARE GREATER OR EQUAL
XX63	SGTL	CGR	2.06.0	COMPARE GREATER
XX64	SEBL	BIS	2.36.0	SET BIT
XX65	REBL	BIR	2.29.0	RESET BIT
XX66	OCOL	OCV	2.38.0	OUTPUT CONVERT
XX67	ICOL	ICV	2.37.0	INPUT CONVERT
XX70	SLEL	CEL	2.06.0	COMPARE EQUAL OR LESS
XX71	SLTL	CLS	2.06.0	COMPARE LESS
XX72	FSUL	FSU	2.28.0	FIELD SUBTRACT
XX73	FADL	FAD	2.28.0	FIELD ADD
XX74	TPDL	TRP	2.05.0	TRANSFER PROGRAM CHARACTERS
XX75	TNDL	TRN	2.04.0	TRANSFER NUMERICS
XX76	TZDL	TRZ	2.03.0	TRANSFER ZONES
XX77	TSDL	TRS	2.02.0	TRANSFER SOURCE CHARACTERS
0100	ILEL	CMX	2.39.0	IN LINE EXIT CHARACTER MODE

CHARACTER MODE OPERATOR INDEX - ALPHABETICAL BY OPERATOR

OPERATOR	MNEMONIC ENG.	MNEMONIC ESPOL	OCTAL CODE	FLOW CHART
BEGIN LOOP	BELL	BNS	XX52	2.19.0
CALL REPEAT FIELD	CLRL	CRF	XX43	2.33.0
COMPARE EQUAL	SEQL	CEQ	XX60	2.06.0
COMPARE EQUAL OR LESS	SLEL	CFL	XX70	2.06.0
COMPARE GREATER	SGTL	CGR	XX63	2.06.0
COMPARE GREATER OR EQUAL	SGEL	CEG	XX62	2.06.0
COMPARE LESS	SLTL	CLS	XX71	2.06.0
COMPARE NOT EQUAL	SNEL	CNE	XX61	2.06.0
END LOOP	ENLL	ENS	XX51	2.20.0
EXIT CHARACTER MODE	RECL	EXC	XX00	2.39.0
FIELD ADD	FADL	FAD	XX73	2.28.0
FIELD ADD (AUX)	FAXL	--	XX33	2.28.0
FIELD SUBTRACT	FSUL	FSU	XX72	2.28.0
FIELD SUBTRACT (AUX)	FXSL	--	XX32	2.28.0
INCREASE TALLY	INTL	INC	XX40	2.35.0
IN LINE EXIT CHAR. MODE	ILEL	CMX	0100	2.39.0
INPUT CONVERT	ICOL	ICV	XX67	2.37.0
JUMP FORWARD CONDITIONAL	CFJL	JFC	XX45	2.22.0
JUMP FORWARD UNCONDITIONAL	FWJL	JFW	XX47	2.22.0
JUMP OUT OF LOOP	JOLL	JNS	XX46	2.21.0
JUMP OUT OF LOOP CONDITIONAL	CJOL	JNC	XX44	2.21.0
JUMP REVERSE CONDITIONAL	CRJL	JRC	XX55	2.23.0
JUMP REVERSE UNCONDITIONAL	REJL	JRV	XX57	2.23.0
OUTPUT CONVERT	OCOL	OCV	XX66	2.38.0
RECALL CONTROL ADDRESS	RPAL	RCA	XX50	2.18.0
RECALL DESTINATION ADDRESS	RDAL	RDA	XX04	2.16.0
RECALL SOURCE ADDRESS	RSAL	RSA	XX53	2.12.0
RESET BIT	REBL	BIR	XX65	2.29.0
SET BIT	SEBL	BIS	XX64	2.36.0
SET DESTINATION ADDRESS	SDPL	SED	XX06	2.14.0
SET SOURCE ADDRESS	SSPL	SES	XX22	2.10.0
SET TALLY	SETL	SEC	XX42	2.35.0

PRINTED IN U.S. AMERICA 10-67

1031986

CHARACTER MODE OPERATOR INDEX - ALPHABETICAL BY OPERATOR (continued)

OPERATOR	MNEMONIC ENG.	MNEMONIC ESPOL	OCTAL CODE	FLOW CHART
SKIP BIT DESTINATION	SBDL	BSD	XX02	2.32.0
SKIP BIT SOURCE	SBSL	BSS	XX03	2.31.0
SKIP FORWARD DESTINATION	FSDL	SFD	XX16	2.25.0
SKIP FORWARD SOURCE	FSSL	SFS	XX31	2.24.0
SKIP REVERSE DESTINATION	RSDL	SRD	XX17	2.26.0
SKIP REVERSE SOURCE	RSSL	SRS	XX30	2.24.0
STORE CONTROL ADDRESS	STPL	SCA	XX54	2.17.0
STORE DESTINATION ADDRESS	STDL	SDA	XX14	2.15.0
STORE SOURCE ADDRESS	STSL	SSA	XX15	2.11.0
STORE TALLY	STAL	STC	XX41	2.34.0
TEST BIT	TEBL	BIT	XX37	2.30.0
TEST FOR ALPHANUMERIC	TANL	TAN	XX36	2.08.0
TEST FOR EQUAL	TEQL	TEQ	XX24	2.07.0
TEST FOR EQUAL OR LESS	TLEL	TEL	XX34	2.07.0
TEST FOR GREATER	TGTL	TGR	XX27	2.07.0
TEST FOR GREATER OR EQUAL	TGEL	TEG	XX26	2.07.0
TEST FOR LESS	TLTL	TLS	XX35	2.07.0
TEST FOR NOT EQUAL	TNEL	TNE	XX25	2.07.0
TRANSFER BLANK FOR NON NUMERIC	TBZL	TBN	XX12	2.05.0
TRANSFER DESTINATION ADDRESS	SDAL	TDA	XX07	2.13.0
TRANSFER NUMERICS	TNDL	TRN	XX75	2.04.0
TRANSFER PROGRAM CHARACTERS	TPDL	TRP	XX74	2.05.0
TRANSFER SOURCE ADDRESS	SSAL	TSA	XX56	2.09.0
TRANSFER SOURCE CHARACTERS	TSDL	TRS	XX77	2.02.0
TRANSFER WORDS	TWDL	TRW	XX05	2.27.0
TRANSFER ZONES	TZDL	TRZ	XX76	2.03.0

CONTROL STATE AND MISCELLANEOUS OPERATORS INDEX - NUMERICAL BY OCTAL CODE

OCTAL CODE	MNEMONIC ENG.	MNEMONIC ESPOL	FLOW CHART	OPERATOR
XXX4 or XXX0	LTST	LITC	3.12.0	LITERAL SYLLABLE
XXX6 or XXX2	OCSL	OPDC	3.10.0	OPERAND CALL SYLLABLE
XXX7 or XXX3	DCSL	DESC	3.10.0	DESCRIPTOR CALL SYLLABLE
0111	PREL	PRL	3.04.0	PROGRAM RELEASE
0211	IINL	ITI	3.03.0	INTERROGATE INTERRUPT
0411	RDTL	RTR	3.02.0	READ TIMER
0435	REWL	XIT	3.11.0	EXIT
1011	COML	COM	3.01.0	COMMUNICATE
2111	IORL	IOR	3.04.0	I/O RELEASE
2211	HP2L	HP2	3.05.0	HALT P2
2411	CHPL	ZPI	3.09.0	CONDITIONAL HALT
3011	SFIL		3.06.0	STORE FOR INTERRUPT
3411	STFL	SFT	3.06.0	STORE FOR TEST
4111	INIL	IPI	3.07.0	INITIATE P1 (SEE NOTE)
4211	PTOL	IP2	3.08.0	INITIATE P2
4411	IOOL	IIO	3.08.0	INITIATE I/O
5111	IFTL	IFT	3.07.0	INITIATE TEST
—	—	—	3.13.0	FETCH
—	SECL	—	3.14.0	SYLLABLE EXECUTION COMPLETE

NOTE:

INITIAL LOAD	3.07.0
INITIATED P2	3.07.0

CONTROL STATE AND MISCELLANEOUS OPERATORS INDEX - ALPHABETICAL BY OPERATOR

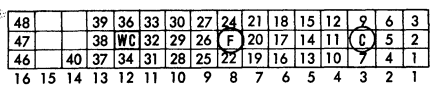
OPERATOR	MNEMONIC ENG.	MNEMONIC ESPOL	OCTAL CODE	FLOW CHART
COMMUNICATE	COML	COM	1011	3.01.0
CONDITIONAL HALT	CHPL	ZPI	2411	3.09.0
DESCRIPTOR CALL SYLLABLE	DCSL	DESC	XXX7 or XXX3	3.10.0
EXIT	REWL	XIT	0435	3.11.0
FETCH			—	3.13.0
HALT P2	HP2L	HP2	2211	3.05.0
INITIATE I/O	IOOL	IIO	4411	3.08.0
INITIATE P1 (SEE NOTE)	INIL	IPI	4111	3.07.0
INITIATE P2	PTOL	IP2	4211	3.08.0
INITIATE TEST	IFTL	IFT	5111	3.08.0
INTERROGATE INTERRUPT	IINL	INI	0211	3.03.0
I/O RELEASE	IORL	IOR	2111	3.04.0
LITERAL SYLLABLE	LTST	LITC	XXX4 or XXX0	3.12.0
OPERAND CALL SYLLABLE	OCSL	OPDC	XXX6 or XXX2	3.10.0
PROGRAM RELEASE	PREL	PRL	0111	3.04.0
READ TIMER	RDTL	RTR	0411	3.02.0
STORE FOR INTERRUPT	SFIL	--	3011	3.06.0
STORE FOR TEST	STFL	SFT	3411	
SYLLABLE EXECUTION COMPLETE	SECL	--	—	3.14.0

NOTE:

INITIAL LOAD	3.07.0
INITIATED P2	3.07.0

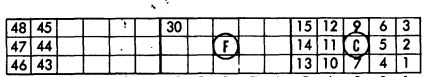
DATA AND CONTROL WORDS

DATA DESCRIPTOR



48 = 1
47 = 0
46 = PRESENCE BIT
40⇒31 = WORD COUNT
30⇒ 16 = DESCRIPTOR'S ADDRESS
15⇒1 = CORE ADDRESS

PROGRAM DESCRIPTOR

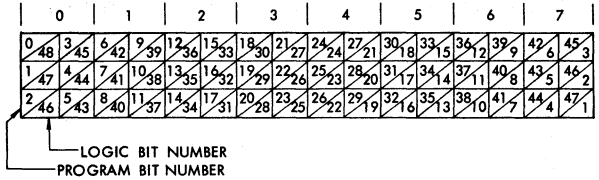


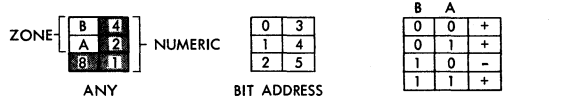
48 = 1
47 = 1
45 = 1
46 = PRESENCE BIT
44 = MODE (1 = CHARACTER MODE)
(0 = WORD MODE)
43 = ARGUMENT BIT
30⇒16 = F (43 = 0)
15⇒ 1 = CORE ADDRESS

PRINTED IN U.S. AMERICA REVISED 8/70

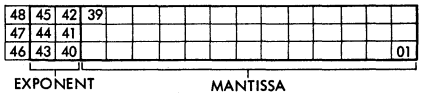
1031986

CHARACTER MODE DATA WORD



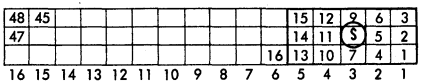


OPERAND



BIT
48 FLAG, "0" FOR OPERAND
47 SIGN OF MANTISSA (0 = +)(1 = -)
46 SIGN OF EXPONENT (0 = +)(1 = -)

INITIATE CONTROL WORD

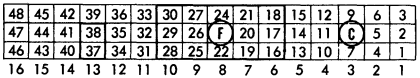


48 = 1
47 = 1
45 = 0
16 = MODE (1 = CHARACTER MODE) (0 = WORD MODE)
15⇒1 = S

FOLLOWING USED FOR TEST INITIATE CONTROL WORD ONLY

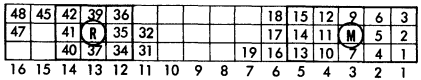
46 = MROF
44 = MWOF
22 = CCCF
Not stored by store for test operator
21 = NCSF
43 ⇒ 35 = Q [9 ⇒ 1] REG
34 ⇒ 29 = Y REG
28 ⇒ 23 = Z REG
20 ⇒ 17 = J REG

INTERRUPT RETURN CONTROL WORD



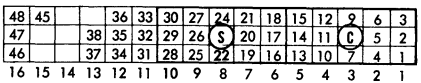
48 = 1
47 = 1
45 = 0
46 = BROF
44 ⇒ 42 = H
41 ⇒ 39 = V
38 ⇒ 37 = L
36 ⇒ 34 = G
33 ⇒ 31 = K
30 ⇒ 16 = F
15 ⇒ 1 = C

INTERRUPT CONTROL WORD



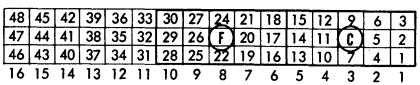
48 = 1
47 = 1
45 = 0
42 ⇒ 34 = R
32 = MARK STACK FF
31 = LEVEL FF (0 = PROGRAM) (1 = SUB PROGRAM)
25 = VARF
19 ⇒ 16 = N
15 ⇒ 1 = M (CHAR. MODE ONLY)

INTERRUPT LOOP CONTROL WORD



48 = 1
47 = 1
45 = 0
46 = AROF
38 ⇒ 37 = L
36 ⇒ 31 = RF
30 ⇒ 16 = S
15 ⇒ 1 = C

RETURN CONTROL WORD



48 = 1
47 = 1
45 = 0
46 = DESCRIPTOR/OPERAND CALL
44 ⇒ 42 = H
41 ⇒ 39 = V
38 ⇒ 37 = L
36 ⇒ 34 = G
33 ⇒ 31 = K
30 ⇒ 16 = F
15 ⇒ 1 = C

PRINTED IN U.S. AMERICA 10-67

1031986

MARK STACK CONTROL WORD

48	45	42	39	36		30	27	24	21	18				
47		41	R	35	32	29	26	F	20	17				
		40	37	34	31	28	25	22	19	16				
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

48 = 1
47 = 1
45 = 0
42 ⇒ 34 = R

32 = MARK STACK FF
31 = LEVEL FF (0 = PROGRAM)
(1 = SUB PROGRAM)
30 ⇒ 16 = F

LOOP CONTROL WORD

48	45			36	33	30	27	24	21	18	15	12	9	6	3
47			38	35	32	29	26	F	20	17	14	11	C	5	2
			37	34	31	28	25	22	19	16	13	10	7	4	1
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

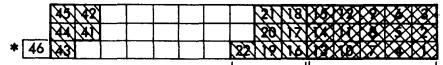
48 = 1
47 = 1
45 = 0

38 ⇒ 37 = L
36 ⇒ 31 = RF
30 ⇒ 16 = F
15 ⇒ 1 = C

SECTION 3
I/O DESCRIPTOR INFORMATION

COMPOSITE RESULT DESCRIPTOR

UNIT DESIGNATE FIELD



* BIT 46
MOD III
FLAG FOR TAPE

ADDRESS FIELD:	Memory address of last access +1 or last access -1 for backward tape read
ERROR FIELD:	See Chart. (Page 3-2)
UNIT DESIGNATE FIELD:	Binary representation of peripheral unit designate number. Refer to UNIT DESIGNATION Chart Page 3-19 for composite listing.

COMMON ERROR FIELD

BIT POS.	ALL UNITS
D16	Designated Unit is Busy
D17	Descriptor Parity Error. Either on Descriptor Address Access (Mem. Cell 10) or Data Descriptor Access.
D18	Designated Unit is not-Ready.
D22	Memory Address Error. Either Memory Overflow or attempt to access Non-existent Memory Address.

UNIT	RESULT DESCRIPTOR LOCATION	FINISHED INTERRUPT LOCATION
I/O-1	14	27
I/O-2	15	30
I/O-3	16	31
I/O-4	17	32

PRINTED IN U.S. AMERICA 10-67 1031986

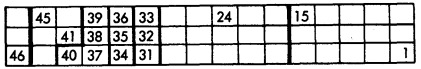
COMPOSITE RESULT DESCRIPTOR (continued)

SPECIAL ERROR FIELD

BIT POS.	TAPE WRITE	TAPE READ	DRUM WRITE	DRUM READ	CARD READ	CARD PUNCH	650 LPM PRINTER	SPO/KEY-BOARD	PAPER READER	PAPER PUNCH
D19	Parity Error from Memory to I/O		Parity Error from Memory to I/O	Parity Error from Drum to I/O	Invalid Character	Parity Error from Memory to I/O	Parity Error from Memory to I/O	Printer-Memory to I/O Parity Error	Parity Error from Reader to I/O	Parity Error from Memory to I/O
D20	Parity Error from tape while writing or Memory Race	Character Parity Error from Tape to I/O Lat. or Long or Memory Race	Memory Race		Read Check Error	Punch Error	Print Check previous line	Keyboard Character, Input Parity Error or Error Button Depressed	Beginning of tape	
D21	End-of-Tape	End-of-File	Designated Drum Channel Locked Out		End-of-File		End of Page (Channel 12 Punch)		End-of-Tape	End-of-Tape

NOTE:
20 and 22
No Write
Ring

MAGNETIC TAPE MOD. III I/O RESULT DESCRIPTOR IN MEMORY



46 = 1 MOD. III DESC.

41 ⇒ 45 = 1 ⇒ 31 (ODD NOS. ONLY) UNITS 1 ⇒ 16

37 = 1 MEM. PARITY

36 = 1 BLANK TAPE

35 = 1 B.O.T.

34 = 1 E.O.T.

33 ⇒ 31 defines the number of characters stored in the last memory address accessed.

Forward Read
- 33 ⇒ 31 = 0 last word complete
- 33 ⇒ 31 = n where "n" equals the number of characters stored in the last partial word

Backward Read
- 33 ⇒ 31 = 7 last word complete
- 33 ⇒ 31 = n where (7-n) equals the number of characters stored in the last partial word.

30 = 1 NO DATA TRANSFER.

26 = 1 BKWD DRIVE.

25	27	
0	0	ALPHA WRITE G.M. END
1	0	ALPHA WRITE W.C. END
1	1	BINARY WRITE W.C. END

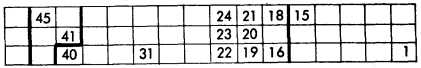
24 = 1 TAPE READ

22 ⇒ 16 SAME AS COMMON ERROR FIELD (SEE NOTE)

1 ⇒ 15 MEMORY ADDRESS OF LAST ACCESS + 1 OR LAST ACCESS -1 FOR BKWD READ

NOTE: 21 is not used during write operations. 40 ⇒ 31 is used as a result descriptor field. 19 and 46 are set if 34, 35, 36 or 37 is set. 37 is set if there is a parity error, memory to I/O Control Unit.

DISK FILE RESULT DESCRIPTOR



48 ⇒ 46 = 0

45 ⇒ 41 = Unit Designate

BCD 6 = DFCU 1
or 14ø

BCD 12 = DFCU 2
or 30ø

40 ⇒ 31 = Remaining Word Count

24 = 1 if Operation was Read, 0 if Operation was Write

23 = 1 if Read Check Error on prior operation

22 = 1 for Core Memory Address Error

21 = 1 if DFEU NOT READY, or an attempt to access non-existent Disk Address (on interrogate, may indicate presence of 40 ms disk)

20 = 1 if PARITY ERROR on transfer of data from Disk to I/O during Read Operation or WRITE LOCKOUT ERROR DURING WRITE

19 = 1 if Core Memory Parity Error; Parity Error during: Disk File Address Transfer, or Data Transfer during Write Operation, to DFCU.

18 = 1 if DFCU NOT READY

16 = 1 if DFCU is busy with another I/O channel

15 ⇒ 1 = last address accessed + 1 for all Read/Write Operations or, initial address + 1 for Read Check and Interrogate Operations.

DATA TRANSMISSION RESULT DESCRIPTOR

46 ⇒ 48 Zero
41 ⇒ 45 Unit Designate (binary 16, 45 ON)
40 = 0 DTC Used
= 1 DTC Not Used
36 ⇒ 39 Terminal Unit number
35 = 0 DCC translator used
= 1 DCC translator not used
31 ⇒ 34 Buffer number
30 = 0 Read or Write operation
= 1 Interrogate operation
28 ⇒ 29 Not used
27 = 0 BCL Code to Internal code translator used
= 1 BCL Code to Internal code translator bypassed
26 Not used
25 = 1 Adapter sensed abnormal condition
23 = 0 Group Mark Ending
= 1 Buffer Exhausted ending or Buffer "filled" ending
22 = 1 Memory overflow
19 = 1 B5500 Memory Parity Error
17 = 1 Parity Error During Descriptor Fetch
16 = 1 DTC Busy
1 ⇒ 15 Core Address

RESULT DESCRIPTOR, READ OR WRITE (30-0)

	21	20	18
Read or Write Completed	0	0	0
Read or Write not Completed (see note)	1	0	0
Read or Write not Completed, Busy	1	1	0
Read or Write not Completed, Not Ready	1	1	1
Read or Write Completed, Busy Flag	0	1	0
Read or Write Completed, Not Ready Flag	0	1	1
DTC Not Ready	0	0	1

Note: Attempt to read a write-ready buffer or attempt to write to a read-ready buffer.

RESULT DESCRIPTOR, INTERROGATE (30 = 1)

	24	21	20	18
IDLE	0	0	0	0
Busy	0	0	1	0
Not Ready	0	0	1	1
Write Ready	0	1	0	0
Read Ready	1	0	0	0
DTC Not Ready	0	0	0	1

Additional states (use, if any, defined by individual adapters)

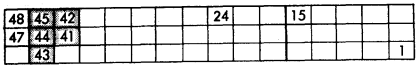
	24	21	20	18
Write Ready, Busy	0	1	1	0
Write Ready, Not Ready	0	1	1	1
Read Ready, Busy	1	0	1	0
Read Ready, Not Ready	1	0	1	1
Read-Write, Ready	1	1	0	0
Read-Write Ready, Busy	1	1	1	0
Read-Write Ready, Not Ready	1	1	1	1

PRINTED IN U.S. AMERICA 10-67

1031986

I/O DESCRIPTORS

SUPERVISORY PRINTER



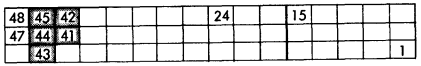
48 = 1, 47 = 0 — ALL I-O DESCRIPTORS

41 ⇒ 45 = 30, 24 = 0 — OUTPUT TO S.P.O.

1 ⇒ 15 = STARTING CORE ADDRESS

Octal Word 5 74 00000 0 00 m aaaa

KEYBOARD



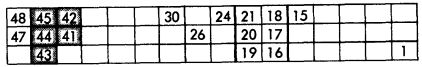
48 = 1, 47 = 0 — ALL I-O DESCRIPTORS

41 ⇒ 45 = 30, 24 = 1 — INPUT FROM KEYBOARD

1 ⇒ 15 = STARTING CORE ADDRESS

Octal Word 5 74 00000 4 00 m aaaa

PRINTER



48 = 1, 47 = 0 — ALL I-O DESCRIPTORS

41 ⇒ 45 = 22 — (PRINTER NO. 1)

41 ⇒ 45 = 26 — (PRINTER NO. 2)

31 ⇒ 35 = — WORD COUNT MOD III I/O

30 = 0 — PERMIT DATA TRANSFER

30 = 1 — INHIBIT DATA TRANSFER

26 = 1 — 132 CHAR. = 0 120 CHAR.

16 ⇒ 19 = 0 — SPACE PAPER PER FORMAT IN 21 & 20

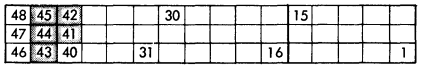
16 ⇒ 19 ≠ 0 — SKIP TO STOP PER 1 ⇒ 11 (PRINTER LOOP TAPE)

1 ⇒ 15 = STARTING CORE ADDRESS

21	20	
0	0	= NO SPACE
0	1	= DOUBLE SPACE
1	0	= SINGLE SPACE
1	1	= DOUBLE SPACE

Octal Word (1) 5 54 000 0 0 0 40 m aaaa
Octal Word (2) 5 64 000 0 0 0 40 m aaaa

DRUM



48 = 1, 47 = 0 — ALL I-O DESCRIPTORS

41 ⇒ 45 = 4 (DRUM NO. 1)

41 ⇒ 45 = 8 (DRUM NO. 2)

46 = 0 — DRUM WRITE

46 = 1 — DRUM READ

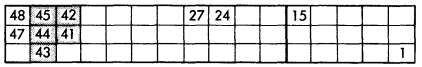
31 ⇒ 40 = WORD COUNT

16 ⇒ 30 = DRUM ADDRESS

1 ⇒ 15 = STARTING CORE ADDRESS

Octal (1) Read 5 10 ccc D aaaa M aaaa
(1) Write 4 10 ccc D aaaa M aaaa
(2) Read 5 20 ccc D aaaa M aaaa
(2) Write 4 20 ccc D aaaa M aaaa

CARD READ



48 = 1, 47 = 0 — ALL I-O DESCRIPTORS

41 ⇒ 45 = 10 (READER NO. 1)

41 ⇒ 45 = 14 (READER NO. 2)

27 = 1 BINARY

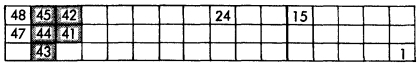
27 = 0 ALPHA

24 = 1 READ

1 ⇒ 15 = STARTING CORE ADDRESS

Octal Word (1) alpha 5 24 0000 0 4 00 M aaaa
(1) binary 5 24 0000 4 4 00 M aaaa
(2) alpha 5 34 0000 0 4 00 M aaaa
(2) binary 5 34 0000 4 4 00 M aaaa

CARD PUNCH



48 = 1, 47 = 0 — ALL I-O DESCRIPTORS

41 ⇒ 45 = 10 (PUNCH NO. 1)

24 = 0 PUNCH

16 = 1 AUX. STACKER

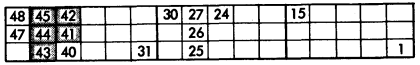
1 ⇒ 15 = STARTING CORE ADDRESS

Octal Word 5 24 00000 0 00 M aaaa

PRINTED IN U.S. AMERICA 10-67

1031986

PAPER TAPE READ OR REWIND

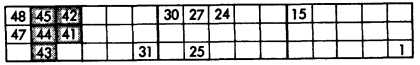


48 = 1 ALL I-O
47 = 0 DESCRIPTORS
41 ⇒ 45 = 18 (READER NO. 1)
41 ⇒ 45 = 20 (READER NO. 2)
31 ⇒ 40 = WORD COUNT (0 = NO OPERATION)
30 = 1 SPACE

27 = 1 BINARY
27 = 0 ALPHA
26 = 0 READ
26 = 1 REWIND
25 = 1 USE WORD COUNTER
24 = 1 READ
1 ⇒ 15 = STARTING CORE ADDRESS

Octal Word (1) 5 44 ccc 4 C 4 00 M aaaa
(2) 5 50 ccc 4 C 4 00 M aaaa

PAPER TAPE PUNCH



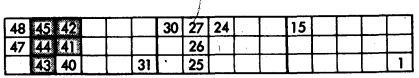
48 = 1 ALL I-O
47 = 0 DESCRIPTORS
41 ⇒ 45 = 18 (PUNCH NO. 1)
41 ⇒ 45 = 20 (PUNCH NO. 2)
31 ⇒ 40 = WORD COUNT (0 = NO OPERATION)

30 = 1 FEED
25 = 1 USE WORD COUNTER
24 = 0 PUNCH
1 ⇒ 15 = STARTING CORE ADDRESS

Octal Word (1) 5 44 ccc 01000 M aaaa
(2) 5 50 ccc 01000 M aaaa

I/O TAPE DESCRIPTORS

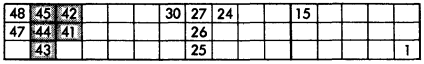
TAPE READ - WORD COUNTER



48 = 1 ALL I-O
47 = 0 DESCRIPTORS
41 ⇒ 45 = 1 ⇒ 31 (ODD NUMBERS ONLY) FOR UNITS 1 ⇒ 16
31 ⇒ 40 ≠ 0 WORD COUNT
30 = 0 DATA TRANSFER
27 = 0 ALPHA
27 = 1 BINARY

26 = 0 FORWARD
26 = 1 BACKWARD
25 = 1 USE WORD COUNTER
24 = 1 READ
1 ⇒ 15 = STARTING CORE ADDRESS

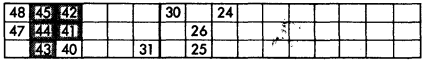
TAPE READ - LONGITUDINAL PARITY GAP



48 = 1 ALL I-O
47 = 0 DESCRIPTORS
41 ⇒ 45 = 1 ⇒ 31 (ODD NUMBERS ONLY) FOR UNITS 1 ⇒ 16
30 = 0 DATA TRANSFER
27 = 0 ALPHA
27 = 1 BINARY

26 = 0 FORWARD
26 = 1 BACKWARD
25 = 0 LONGITUDINAL PARITY GAP END
24 = 1 READ
1 ⇒ 15 = STARTING CODE ADDRESS

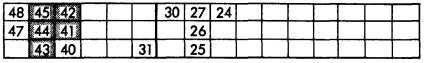
TAPE SPACE - NORMAL



48 = 1 ALL I-O
47 = 0 DESCRIPTORS
41 ⇒ 45 = 1 ⇒ 31 (ODD NUMBERS ONLY) UNITS 1 ⇒ 16
30 = 0 NORMAL SPACE

26 = 0 FORWARD
26 = 1 BACKWARD
25 = 1 USE WORD COUNTER
24 = 1 READ
WC = 0 SPACE 1 RECORD

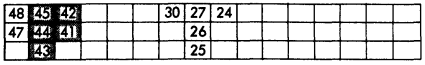
TAPE SPACE - MAINTENANCE



48 = 1 ALL I-O
47 = 0 DESCRIPTORS
41 ⇒ 45 = 1 ⇒ 31 (ODD NUMBERS ONLY) UNITS 1 ⇒ 16
30 = 1 MARK TIME FOR MAINTENANCE PURPOSES.
27 = 0 SPACE FORWARD 2 RECORDS AND MARK INTER-RECORD GAP.
27 = 1 SPACE AND MARK TIME TO VALID RECORD.

26 = 0 FORWARD
26 = 1 BACKWARD
25 = 1 USE WORD COUNTER
WC = 0 SPACE
24 = 1

TAPE REWIND

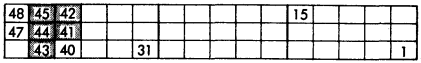


48 = 1 ALL I-O
47 = 0 DESCRIPTORS
41 ⇒ 45 = 1 ⇒ 31 (ODD NUMBERS ONLY) UNITS 1 ⇒ 16
30 = 1 NO DATA TRANSFER

27 = 0 ALPHA
26 = 1 BACKWARD
25 = 0 DO NOT USE WORD COUNTER
24 = 0 WRITE

PRINTED IN U.S. AMERICA REVISED 8/70 1031966

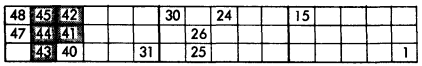
TAPE WRITE



48 = 1 ┐ ALL I-O
47 = 0 ┘ DESCRIPTORS
41 ⇒ 45 = 1 ⇒ 31 (ODD NUMBERS ONLY) UNITS 1 ⇒ 16
30 = 0 DATA TRANSFER
27 = 0 ALPHA
27 = 1 BINARY

26 = 0 FORWARD
25 = 1 USE WORD COUNTER
24 = 0 WRITE
WC ≠ 0 BINARY WRITE MOD II & III
ALPHA WRITE MOD III
1 ⇒ 15 = STARTING CORE ADDRESS

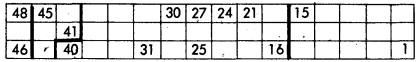
TAPE ERASE



48 = 1 ┐ ALL I-O
47 = 0 ┘ DESCRIPTORS
41 ⇒ 45 = 1 ⇒ 31 (ODD NUMBERS ONLY) UNITS 1 ⇒ 16
30 = 1 NO DATA TRANSFER
27 = 0 ALPHA
27 = 1 BINARY

26 = 0 FORWARD
25 = 1 USE WORD COUNTER
24 = 0 WRITE
WC ≠ 0 BINARY ERASE MOD II & III
ALPHA ERASE MOD III
1 ⇒ 15 = STARTING CORE ADDRESS

DISK FILE DESCRIPTORS AND OPERATION



48 = Flag Bit; 1 if Descriptor

46 = Presence Bit; 1 if Core Memory assigned

45 ⇒ 41 = Unit Designate

BCD 6 = DFCU 1
or 14ø (ø = octal)

BCD 12 = DFCU 2
or 30ø

40 ⇒ 31 = Word Count (Values of 0000 - 1777ø)

30 = 1 Read Check - Inhibit Data Transfer

27 = 1 for Binary, 0 for Alpha (BCL) translation

25 = 1 to use Word Counter Override

24 = 1 for Disk File Read, 0 for Disk File Write

21 ⇒ 16 = Number of Segments (Values of 00 - 77ø, where 77ø = 63 segments)

15 ⇒ 1 = Core Memory Address**

**NOTE:

Last seven (7) characters of first word addressed by 15 - 1 contain Disk File Address; first character is not used.

PRINTED IN U.S. AMERICA 10-67 1031986

DESCRIPTOR COMBINATIONS

40 → 31 WORD COUNT	30	27	25	24	21 → 16 SEGMENT COUNT "n"	OPERATION
	1				1 ≤ n ≤ 77ø	READ CHECK
		0	0	1	1 ≤ n ≤ 77ø	Read with BCL translation; ignore Word Count
		1	0	1	1 ≤ n ≤ 77ø	Read without translation (Binary); ignore Word Count
1 ≤ WC ≤ 1777ø		0	1	1	1 ≤ n ≤ 77ø	Read with BCL translation; Word Count Override
1 ≤ WC ≤ 1777ø		1	1	1	1 ≤ n ≤ 77ø	Read without translation; Word Count Override
		0	0	0	1 ≤ n ≤ 77ø	Write with BCL translation; ignore Word Count
		1	0	0	1 ≤ n ≤ 77ø	Write without translation; ignore Word Count
1 ≤ WC ≤ 1777ø		0	1	0	1 ≤ n ≤ 77ø	Write with BCL translation; Word Count Override
1 ≤ WC ≤ 1777ø		1	1	0	1 ≤ n ≤ 77ø	Write without translation; Word Count Override
WC = 0			1			Interrogate

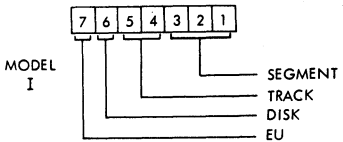
NOTE: The "0" and "1" are required where shown, and blanks are irrelevant.

DISK FILE CHART

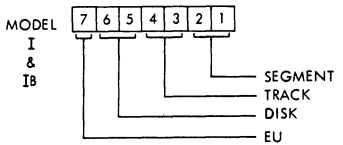
MOD.	DISK	MODEL I 240 C/S OPTION	MODEL IB 240 C/S OPTION
1	1	0000000 0009999	0000000 0040000 0009999 0049999
	2	0010000 0019999	0010000 0050000 0019999 0059999
	3	0020000 0029999	0020000 0060000 0029999 0069999
	4	0030000 0039999	0030000 0070000 0039999 0079999
2	5	0040000 0049999	0080000 0120000 0089999 0129999
	6	0050000 0059999	0090000 0130000 0099999 0139999
	7	0060000 0069999	0100000 0140000 0109999 0149999
	8	0070000 0079999	0110000 0150000 0119999 0159999
3	9	0080000 0089999	0160000 0200000 0169999 0209999
	10	0090000 0099999	0170000 0210000 0179999 0219999
	11	0100000 0109999	0180000 0220000 0189999 0229999
	12	0110000 0119999	0190000 0230000 0199999 0239999
4	13	0120000 0129999	0240000 0280000 0249999 0289999
	14	0130000 0139999	0250000 0290000 0259999 0299999
	15	0140000 0149999	0260000 0300000 0269999 0309999
	16	0150000 0159999	0270000 0310000 0279999 0319999
5	17	0160000 0169999	0320000 0360000 0329999 0369999
	18	0170000 0179999	0330000 0370000 0339999 0379999
	19	0180000 0189999	0340000 0380000 0349999 0389999
	20	0190000 0199999	0350000 0390000 0359999 0399999

PRINTED IN U.S. AMERICA REVISED 8/70. 1031986

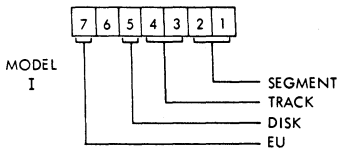
96 OPTION



240 OPTION



480 OPTION



STANDARD INPUT MESSAGE FROM ANSWER-BACK DRUM ON ASR STATIONS

ASCII:	CR	LF	RO	(1 → 12)	CR	LF	X-ON
BCL:	DISC	DISC	DISC	(1 → 12)	DISC	DISC	←

NOTE:

1. On input the 980 adapter replaces X-ON with GM character or an ASCII (!) to a disconnect.
2. Input data from paper tape must have the following format: (DATA---------- X-ON ← RO).
3. All control functions are discarded.
4. The operator sends a backspace character to effectively erase the last character transmitted. The number of backspaces sent by the operator erases a corresponding number of characters. Hardware does not allow backspacing beyond the beginning of the B487 buffer.
5. The operator depresses Control L (form out) to erase the contents of the entire buffer associated with that station. This is a quick way to "backspace" to the beginning of the buffer.

When using the Model 33/35 Teletype typewriters, five of the 64 BCL characters are reserved for special functions on output.

B487

BCL	ASCII	OUTPUT
≥	!	TRANSLATED TO DISCONNECT BY 980 ADAPTER
≤	'	TRANSLATED TO CARRIAGE RETURN BY B487
≠	†	TRANSLATED TO LINE FEED BY B487
>	>	TRANSLATED TO X-ON BY 980 ADAPTER
<	<	TRANSLATED TO RUB OUT BY B487
←	←	END OF MESSAGE (not transmitted)

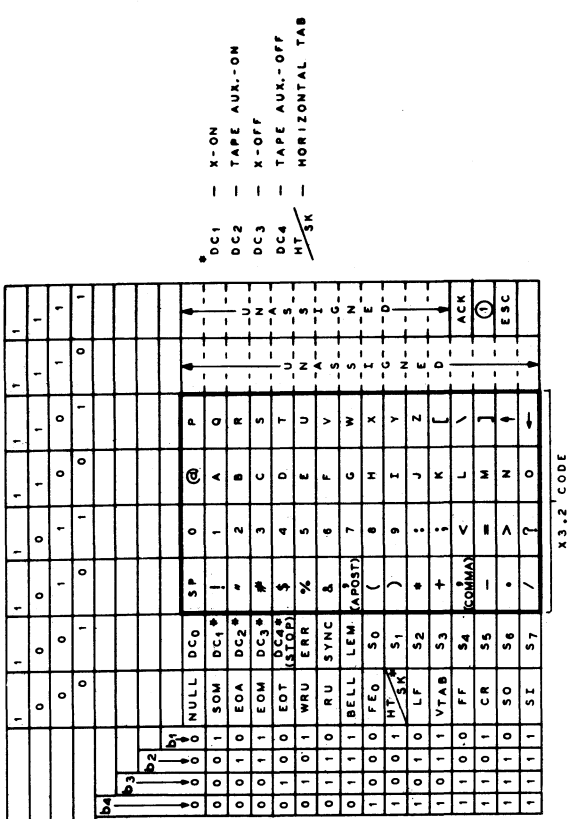
PRINTED IN U.S. AMERICA 10-67

1031986

STANDARD OUTPUT REPLY TO WRU:

BCL:	≤	≠	<	(1 → 12 characters)	≤	≠	>	→
ASCII:	CR	LF	RO	(1 → 12 characters)	CR	LF	X-ON	GM

AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE



TWX/TYPEWRITER ADAPTER INTERRUPT CHART

	BCL	ASCII	FUNCTION	INTERRUPT	CONDITION	COMMENT
INPUT	?	? !	FORCED ERROR	NONE	A	MCP FLAG-CONTROL MESSAGE, (PLUGGABLE OPTION B5500 ONLY)
	IFAL		EOS	X	RR	FULL BUFFER
	GM	<-/X-ON	EOM	X	RR	END-OF-MESSAGE
	≥	!	EOT	X	RR A	TERMINATE CALL
		LONG SPACE	LOSS OF CARRIER DISCONNECT	X	IDLE A	TERMINATE CALL, RESET BUFFER ADD
		LONG SPACE	BREAK	NONE	IDLE	RESET BUFFER ADD
		CONTROL L	FORM OUT	NONE	BUSY	RESET BUFFER ADD
		<	BACKSPACE	NONE	BUSY	DELETE LAST CHARACTER
		CONTROL E	WRU	X	WRA	INITIATE WRU RESPONSE
		BUFFER OVERFLOW	OPERATOR ERROR	FORCED X	BUSY	ERROR INDICATION CHARACTER LOST
		CARRIER	INITIAL DETECTION	X	WRA	INITIATE WRU RESPONSE
OUTPUT	IFAL		EOS	X	WR	EMPTY BUFFER
	GM		EOM	X	IDLE	END-OF-MESSAGE
	≥	!/EOT	EOT	X	IDLE A	TERMINATE CALL
		LONG SPACE	LOSS OF CARRIER DISCONNECT	X	IDLE A	TERMINATE CALL
		LONG SPACE	BREAK	X	RR A	TERMINATE CALL
	≤	CR	CR	NONE	BUSY	TRANSLATE (↑) TO (CR)
	≠	LF	LF	NONE	BUSY	TRANSLATE (↓) TO (LF)
	>	X-ON	START PAPER TAPE	NONE	BUSY	PAPER TAPE OPTION
	<	RO	RUB-OUT	NONE	BUSY	TRANSLATE (<) TO (RO)

PRINTED IN U.S. AMERICA 10-67 1031986

DATA TRANSMISSION REGISTERS

"D" Register

	Address	Control
DA6	N/A	Int.
5	X8	N/A
4	X4	WR
3	X2	RR
2	X1	Busy
1	Z1	Not Ready

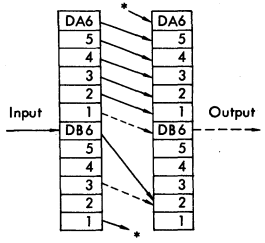
DB6	Busy	Busy
5	Y8	
4	Y4	
3	Y2	
2	Y1	
1	Abn.	Abn.

DA6	DB6	
0	0	Idle
0	1	In use by Adapter
1	0	Interrupt
1	1	In use by System

"M" Register

M=0 No Memory Cycle
M=1 Buffer Proper
M=2 Address or Control Cell
M=3 I/O Cell

"D" Register Shift



"N" Register

N=0 No System Activity
N=1 System Sync
N=2 Scan Cycle
N=3 Not Used
N=4 Read to GM
N=5 Read - Ignore GM
N=6 Write to GM
N=7 Write - Ignore GM

"L" Register

L1 = Buffer Final Location
L2 = Operation Complete
L4 = Temp. Storage for Abnormal & Input Information Bits

UNIT DESIGNATION

UNIT		UNIT NAME	$\overline{D24F}$	D24F	**
A	02	MAGNETIC TAPE	WRITE	READ	1
B	06	MAGNETIC TAPE	WRITE	READ	2
C	12	MAGNETIC TAPE	WRITE	READ	3
D	16	MAGNETIC TAPE	WRITE	READ	4
E	22	MAGNETIC TAPE	WRITE	READ	5
F	26	MAGNETIC TAPE	WRITE	READ	6
H	32	MAGNETIC TAPE	WRITE	READ	7
J	36	MAGNETIC TAPE	WRITE	READ	8
K	42	MAGNETIC TAPE	WRITE	READ	9
L	46	MAGNETIC TAPE	WRITE	READ	10
M	52	MAGNETIC TAPE	WRITE	READ	11
N	56	MAGNETIC TAPE	WRITE	READ	12
P	62	MAGNETIC TAPE	WRITE	READ	13
R	66	MAGNETIC TAPE	WRITE	READ	14
S	72	MAGNETIC TAPE	WRITE	READ	15
T	76	MAGNETIC TAPE	WRITE	READ	16
2	04	UNASSIGNED			-
4	10	#1 DRUM	WRITE	READ	17
6	14	#1 DISK FILE	WRITE	READ	19
8	20	#2 DRUM	WRITE	READ	18
10	24	#1 CARD	PUNCH	READER	23/24
12	30	#2 DISK FILE	WRITE	READ	20
14	34	#2 CARD READER		UNASSIGNED	25
16	40	DATA TRANS.	WRITE	READ	31
18	44	#1 PAPER TAPE	PUNCH	READER	27/28
20	50	#2 PAPER TAPE	PUNCH	READER	30/29
22	54	#1 PRINTER		UNASSIGNED	21
24	60	UNASSIGNED		UNASSIGNED	-
26	64	#2 PRINTER		UNASSIGNED	22
28	70	UNASSIGNED		UNASSIGNED	-
30	74	SUPERVISORY	PRINTER	KEYBOARD	26

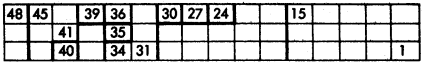
** INTERROGATE PERIPHERAL STATUS OPERATOR IPSL (2431) WILL SET THIS BIT IF THE UNIT IS READY.

INTERROGATE I/O STATUS OPERATOR TIOL (6431) WILL SET A REGISTER TO THE FOLLOWING CONFIGURATION:

I/01 - $A01F \cdot \overline{A02F} \cdot \overline{A03F}$
I/02 - $\overline{A01F} \cdot A02F \cdot \overline{A03F}$
I/03 - $A01F \cdot A02F \cdot \overline{A03F}$
I/04 - $\overline{A01F} \cdot \overline{A02F} \cdot A03F$

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

DATA TRANSMISSION DESCRIPTOR



48 ➡ 46 Flags

41 ➡ 45 Unit Designate (Binary 16, 45 = 1).

40 = 0

36 ➡ 39 Terminal unit number (1 ➡ 15)
Zero = DTC and DTTU designated terminal and buffer.

35 = 0 Terminate buffer loading or unloading when a group mark is detected or buffer exhausted.

35 = 1 Ignore group marks.

31 ➡ 34 Buffer number

30 = 1 Interrogate (24 must be zero)

30 = 0 Read or Write operation

28 ➡ 29 Not Used

27 = 1 Bypass BCL code to internal code translator

27 = 0 Use the BCL code to internal code translator

25 ➡ 26 Not Used

24 = 1 Read

24 = 0 Write or Interrogate

16 ➡ 23 Not Used

1 ➡ 15 Core Memory Address

SECTION 4

MESSAGES

GENERAL

The operator and the MCP communicate with each other by means of the supervisory printer and keyboard. Through the use of the supervisory printer, the MCP can direct the operator and supply the answers to inquiries from him. The operator, on the other hand, can acknowledge instructions typed by the MCP and initiate inquiries that must be answered by the MCP.

SYSTEM MESSAGES

The messages given to the operator are of two basic types: those for informative purposes only and those requiring action by the operator. To minimize the amount of time used by the supervisory printer, the messages are made up of mnemonic codes followed by the variable information that is needed to make the message meaningful. Each element of the message (including the mnemonic code) will be separated from adjacent elements by at least one blank.

A system message which requires an action by the system operator is prefixed with the character #.

System messages which denote that a program will be discontinued before EOJ are preceded by the character -.

System messages related to the breakout and restart facility are preceded by the character pair --.

In the descriptions of system messages, the construct < job specifier> will be used and is defined as follows: <program specifier> = <mix index>. An example of a <job specifier> is: PROGID/SUPID=1.

The <mix index> provided in a <job specifier> is one to be used in any keyboard input messages referencing the subject program if the input message requires a <mix index>.

Another construct which will be used in describing keyboard output messages is <terminal reference>. The <terminal reference> is defined as: S = <integer>, A = <integer>, where the <integer> following the S is the number of the program segment which was being executed when the subject program was discontinued (except in the case of an intrinsic segment where the number refers to the last non-intrinsic segment executed), and the <integer> following the A is the relative address, within the segment specified, of the syllable that was last executed.

A third construct is <file specifier>. A <file specifier> is defined as: <file identification prefix>/<file identification> or <program identifier>/<program identifier suffix>.

A complete list of system messages is presented below in alphabetical order, together with an explanation of each message and any operator action that is required. In addition, the items of the messages specified by the construct <unit mnemonic> will actually appear in mnemonic form. They are:

© 1970 Burroughs Corporation

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

MTA	Magnetic Tape Unit A	MTT	Magnetic Tape Unit T
MTB	Magnetic Tape Unit B	LPA	Line Printer A
MTC	Magnetic Tape Unit C	LPB	Line Printer B
MTD	Magnetic Tape Unit D	CPA	Card Punch A
MTE	Magnetic Tape Unit E	CRA	Card Reader A
MTF	Magnetic Tape Unit F	CRB	Card Reader B
MTH	Magnetic Tape Unit H	PRA	Paper Tape Reader A
MTJ	Magnetic Tape Unit J	PRB	Paper Tape Reader B
MTK	Magnetic Tape Unit K	PPA	Paper Tape Punch Unit A
MTL	Magnetic Tape Unit L	PPB	Paper Tape Punch Unit B
MTM	Magnetic Tape Unit M	SPO	Supervisory Printer
MTN	Magnetic Tape Unit N	CDA	Pseudo Card Reader A
MTP	Magnetic Tape Unit P	CDB	Pseudo Card Reader B
MTR	Magnetic Tape Unit R	CDC	Pseudo Card Reader C
MTS	Magnetic Tape Unit S	CDD	Pseudo Card Reader D

KEYBOARD OUTPUT MESSAGES

The keyboard output messages or system messages are listed and described in the following paragraphs.

<job specifier> = <mix index> ACCEPT

This message is typed on the SPO for a programmatic operator input request.

-ACTV INTRGT TU 0 <prog. id.> / <prog. id.> = <mix index>

This message notifies the operator that a program attempting an active interrogate on terminal unit 0 has terminated.

BED OVRFLW

The occurrence of this message denotes that too many entries have been made in the BED, a table used by the control section of the MCP. If the condition indicated by this message should occur, a HALT-LOAD operation is required.

<job specifier> = <mix index> BOJ <time> FROM <remote specifier>

This message is typed when an object program first begins to execute, providing the TYPE BOJ option is set. The FROM <remote specifier> will be included if the job was executed from a remote station with SPO capabilities.

<compiler name> / <program identifier> = <mix index> BOJ <time> FROM <remote specifier>

This message is typed when either the ALGOL or the COBOL Compiler begins a compilation, providing the TYPE BOJ option is set. The FROM <remote specifier> will be included if the compile was executed from a remote station with SPO capabilities.

<unit mnemonic> BUSY

The occurrence of this message denotes that an I/O operation was attempted on the specified unit, and the unit was found to be apparently busy.

CONTROL CARD ERROR <unit mnemonic> information from control card

The occurrence of this message indicates that the MCP has expected to read control information from the designated I/O unit but has found the information to be in error.

CP RQD <data file designator> <rdc> : <job specifier>

The occurrence of this message indicates that a program has need for a card punch and no such I/O device is currently available.

DATACOM / INQUIRY INTERRUPT IGNORED BY MCP

This message is typed on the SPO if a datacom interrupt is received and the MCP has been compiled with the DATACOM and INQUIRY options set FALSE.

<unit mnemonic> / <I/O operation> DA = <integer>; #SEG = <integer>; #RTRY = <integer>; #TRNS = <integer>

This message is typed when retries had to be made on the disk file. The <I/O operator> is an R if it was on a read, W if on a write. The number appearing after DA is the disk address; the number appearing after #SEG is the number of segments read or written, the number appearing after #RTRY is the number of retries necessary (modulo 10). If this number = 0, then a successful retry was not made. The number appearing after #TRNS is the number of disk transactions since the last HALT-LOAD operation.

-DC TU NOT OUTPUT POSSIBLE <job specifier> . <termination reference>

The occurrence of this message denotes that an object program attempted to perform a write on a terminal unit that was not set for output. Because of this erroneous action, processing of the object program was discontinued.

-DEC ERR : ARRAY DIMENSION = <integer>,

<job specifier> , <terminal reference> will be typed if an array with an illegal row size is declared.

-DEC ERR : NO. DISK ROWS = <integer> ,

<job specifier> , <terminal reference> will be typed if a disk file is declared with more than 20 rows.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

DECK <integer> REMOVED

This message is typed when a control deck is removed from the disk because of the completion of the job or a keyboard input message.

DISK FAILURE - <unit mnemonic>

If this message occurs and is not followed by a ...#RTRY=... message, then a HALT-LOAD must be performed.

DKA(R)bDA = nnnnnnnn; b#SEG = mm; b#RTRY = x; b#TRNSb = byyyyyyyyyy
DKA(N)bDA = nnnnnnnn; b#SEG = mm; b#RTRY = x; b#TRNSb = byyyyyyyyyy
DKB(R)bDA = nnnnnnnn; b#SEG = mm; b#RTRY = x; b#TRNSb = byyyyyyyyyy
DKB(W)bDA = nnnnnnnn; b#SEG = mm; b#RTRY = x; b#BRNSb = byyyyyyyyyy

n = absolute disk address in decimals.
m = number of segments in the I/O operation.
x = number of retries required before successful completion: 1-9; 0 indicates that a successful retry was not accomplished.
y = number of read/writes in n area since the last Halt/Load.

This message indicates an area on disk was accessed which caused the I/O error routines to attempt retries. This message generally will occur in conjunction with the DISK FAILURE message. Information in this message should be retained for analysis by the Field Engineers.

-DIV BY ZERO <job specifier> , <termination reference>

The occurrence of this message denotes that an object program performed a Divide operation using a zero denominator. Consequently, processing of the subject program was discontinued.

DIV BY ZERO BRANCH < job specifier> , <termination reference>

This message will be typed upon the occurrence of a Divide by Zero when the programmatic recovery feature is being used.

<job specifier> = <mix index> DS-ED

This message is typed if processing of an object program is discontinued before End-of-Job, providing the EOJ option is set.

<compiler name> / <program identifier> = <mix index> DS-ED

This message is typed if a compilation is discontinued before the compiler has reached End-of-Job, providing the TYPE EOJ option is set.

DT PLEASE

This message is typed at HALT-LOAD time if the TYPE DATE option has been set. The system operator is required to enter a DT message before processing can commence.

DUMP REMOTE/LOG

This message will be typed when the log is half-full.

<file specifier> DUMPED

This message is typed after the MCP has performed an operation specified on a DUMP control card.

DUP FIL < data file designator> <rdc> : <job specifier> <duplicate file list>

The occurrence of this message denotes that an object program wishes to open an input file and that the MCP has found more than one file with the desired identification. Files on disk are not taken into regard. The duplicate-file condition causes the designated program to be suspended until operator action is taken. The condition may be rectified by making only one of the acceptable files available and then entering a <mix index> OK message.

ADDITIONAL USE OF THE IL MESSAGE

The IL message may now be used to designate the file to be opened when the DUP FILE condition arises.

DUP LIBRARY <file specifier> : <job specifier>

The occurrence of this message indicates that an attempt has been made to add a file to the disk library, but the file's name is identical to the name of a file already in the disk directory. The program which attempted to add the file to the library is temporarily suspended until the operator remedies the situation. To remedy the situation, the system operator may eliminate the conflict by using a CHANGE card or REMOVE card and then an OK message, or he may DS the program that attempted to place the new file in the library, or he may enter an RM keyboard input message.

-EOF NO LABEL <file designator> : <job specifier> , <termination reference>

The occurrence of this message denotes that an object program has reached the end of the designated input file and has not specified what is to be done. Consequently, processing of the program was discontinued.

<job specifier> = mix index> EOJ

This message is typed when an object program reaches End-of-Job, providing the TYPE EOJ option is set.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

<compiler name> / <program identifier> = <mix index> EOJ

This message is typed when a compiler reaches End-of-Job, providing there were no syntax errors and providing the TYPE EOJ option was set.

-EOT NO LABEL <file designator> : <job specifier> , <termination reference>

The occurrence of this message denotes that an object program has reached the end of the designated file's declared area, as on disk. Consequently, processing of the program was discontinued.

-EXCESS TIME <job specifier> , <termination reference>

The occurrence of this message denotes that the process time of an object program has exceeded the time specified on its PROCESS program-parameter card. Consequently, processing of the program was discontinued.

<file specifier> EXPIRED

This message is typed in reference to files on disk in response to the EX input message if (the file's date of last access) + (the file's SAVE factor) does not result in a date greater than the current date.

-EXPON OVRFLW <job specifier> , <termination reference>

The occurrence of this message denotes that an object program has performed an operation which caused an exponent overflow to occur. Consequently, processing of the program was discontinued.

EXPON OVRFLW BRANCH <job specifier> , <termination reference>

This message will be typed upon the occurrence of an exponential overflow when the programmatic recovery feature is being used.

FACTOR = x, MAX CORE = y, USING z

The MCP will respond to a TF message with this message. The letter x is the Factor itself, y is the actual number of core cells available to object programs, multiplied by the Multiprocessing Factor, and z is the sum of the core requirements of the jobs actually running.

<unit mnemonic> <read-write flag> FAILURE - D <integer>

This message indicates that one of the following error conditions persisted after ten retries:

<integer> = 19 - parity error between I/O control and core or disk file control.

<integer> = 20 - parity error on transfer from disk.

-FILE UNOPENED <job specifier> , <terminal reference>

currence of this message denotes that an object program attempted to write on a file that has not been opened. Consequently, processing of the program was discontinued.

-FLAG BIT <job specifier> , <terminal reference>

The occurrence of this message denotes that an object program has performed an operation which caused a word with a flag bit of 1 to be accessed as if it were an operand. Consequently, processing of the program was discontinued.

FLAG BIT BRANCH <job specifier> , <terminal reference>

This message will be typed upon the occurrence of a flag bit when the programmatic recovery feature is being used.

FM RQD <data file designator> <rdc> : <job specifier>

The occurrence of this message indicates that a program is ready to open a file which -- as specified on a label equation card -- is required to use special forms. The FM message must be entered before the subject program can continue processing.

-FMT ERR NO LBL <job specifier> , <terminal reference>

The occurrence of this message denotes that the object program tried to evaluate the editing phrase of an ALGOL format using a negative or undefined list element to evaluate a dynamic format phrase. No action label was supplied. Consequently, processing of the program was discontinued.

<unit mnemonic> IN <data file designator> <rdc> : <job specifier>

This message is typed when a program opens a card or tape file for input, providing the necessary options have been set. The message will be typed for object program files if the TYPE OPN option is set. The message will be typed for compiler files if both the TYPE OPN and TYPE CMPLRFIL options are set.

-INTGR OVRFLW <job specifier> , <termination reference>

The occurrence of this message denotes that an object program performed an operation which caused an integer overflow to occur. Consequently, processing of the program was discontinued.

INTGR OVRFLW BRANCH <job specifier> , <termination reference>

This message will be typed upon the occurrence of an integer overflow when the programmatic recovery feature is being used.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

-INVALD ADRSS <job specifier> , <termination reference>

The occurrence of this message denotes that an object program performed an operation which addressed a memory location in an absent memory module or an address less than 00512. Consequently, processing of the program was discontinued.

INVALD ADRSS

The occurrence of this message denotes that an invalid address occurred during processing in control state, and the invalid address could not be associated with a particular program in the MIX. If the condition indicated by this message should occur, a HALT-LOAD operation is required.

INVALD LINK

The occurrence of this message denotes that an invalid memory link was detected by the MCP. If the condition indicated by this message should occur, an H/L operation is required.

INVALID ARG CONCAT <job specifier> , <terminal reference>

This message announces that execution of the FORTRAN intrinsic function CONCAT is terminated if the necessary conditions are not satisfied. The function provides general partial word facilities and is referenced as follows:

CONCAT (A, B, S1, S2, N)

where A and B are any integer or real expressions; S1, S2, and N are integer expressions. S1, S2, and N must satisfy the following conditions:

S1 > 0
S2 > 2
N > 0
S1 + N < 48
S2 + N $\leq$ 48

If these conditions are not satisfied, execution is terminated.

The value of the function is the concatenation of A and B as specified by S1, S2, and N. N bits are taken from B, starting as bit 52, and replace N bits of A, starting at bit S1. This has the same effect as the ALGOL expression A and B S1:S2 = N , assuming S1, S2, and N could be arbitrary expressions in ALGOL. The following ALGOL constructs can be performed with CONCAT:

ALGOL	FORTRAN
C. 12:6 "X";	C = CONCAT (C, "X", 12, 42, 6)
C A&B 18:45:3 ;	C = CONCAT (A, B, 18, 45, 3)
C B. 36:6	C = CONCAT (C, B, 42, 36, 6)

Note that, just as in ALGOL, bit 0 (the flag bit) cannot be included in either field.

-INVALID EOJ <job specifier> , <termination reference>

The occurrence of this message denotes that a COBOL program attempted to execute the END-OF-JOB statement. Consequently, processing of the program was discontinued.

-INVALID INDEX <job specifier> , <terminal reference> , EFF INX IS - <index value> .

The occurrence of this message denotes that an object program attempted to index out of the limits (in a negative direction) of the array being referenced. Processing of the program was discontinued. <index value> has a maximum size of eight digits.

-INVALID INDEX <job specifier> , <terminal reference> , <index value> GEQ <descriptor size field>

The occurrence of this message denotes that an object program attempted to index out of the limits (in a positive direction) of the array being referenced. Processing of the program was discontinued. <index value> has a maximum size of four digits.

<description size field> has a maximum size of four digits.

INVALID INDEX BRANCH <job specifier> , <termination reference>

This message will be typed upon the occurrence of an invalid index when the programmatic recovery feature is being used.

-INVALID PRL <job specifier> , <terminal reference>

The occurrence of this message indicates that either:

a. The DSKTOG (OPTN 28) was set and an object program tried to access an absolute disk address below the user disk area; or

b. The RELTOG (OPTN 27) was set and an ALGOL object program attempted to perform a RELEASE statement referencing disk.

In either case, processing of the program was discontinued.

<unit mnemonic> INV CHR IN COL <integer>

This message is typed when a card has an invalid character other than one in column 1 of a control card. The column with the invalid character is given in the message. The operator must replace this card with a correct card.

INV KBD { typed-in information }

This message is typed if the MCP does not recognize a message entered from the keyboard.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

-INV STN <tu> <buff>

This message is typed if the MCP does not recognize the remote station address used in a remote terminal message.

I/O ERROR <integer> <file designator> : <job specifier>

There are a number of messages which have the above format; the <integer> in the message denotes its specific meaning. The meanings of this message are listed below according to the <integer>:

<integer> value	Meaning
1	A COBOL program attempted to open an input file that was not closed; consequently, processing of the program was discontinued.
3	A COBOL program attempted to open reverse a file that was not closed; consequently, processing of the program was discontinued.
5	A COBOL program attempted to open reverse a file that was not blocked properly; consequently, processing of the program was discontinued.
6	A COBOL program attempted to open an output file that was not closed; consequently, processing of the program was discontinued.
11	An attempt was made to close an input file which was closed or never opened.
12	An attempt was made to close an output file which was closed or never opened.
15	An attempt was made to read a file for which AT END has already been processed.
16	The record count on an input tape does not agree with the internally accumulated record count. The external record or block count is printed out first in the error message; then the internal record or block count is printed.
17	The block count on an input tape does not agree with the internally accumulated block count. The external record or block count is printed out first in the error message; then the internal record or block count is printed.
18	The HASH TOTAL on a COBOL input tape does not agree with the internally accumulated HASH TOTAL.
19	An irrecoverable parity error has occurred during reading of a file assigned to disk or tape. The message will be typed once for each block which is in error unless a USE procedure has been specified. The USE procedure (if any) will be executed and control will be transferred to the statement following the READ statement.

<integer> value	Meaning
20	An irrecoverable parity error occurred on an output tape or disk file. The USE procedure has been executed allowing programmatic closing of files which must be saved and the program is now being DS-ED.
21	An attempt was made to READ from a file opened as OUTPUT. The program is DS-ED.
22	An attempt was made to read from a row of a disk file which was never created. To get this error, the record number must be less than the highest record number written and greater than 1. When a RANDOM file is written, but records fall only in rows 1 and 3 of a 3 row file, attempts to access records in row 2 will cause I/O ERROR 22 instead of executing INVALID KEY statements. A row of disk space will be assigned and the appropriate record made available to the using program. The contents of the record made available will of course be unpredictable.
23	An attempt was made to WRITE on a file which was opened as INPUT. The program is DS-ED.
24	An attempt was made to WRITE on a file which was opened REVERSED. The program is DS-ED.
25	Improper code was passed to the COBOLIO intrinsics. The program is DS-ED.
26	A block of less than 8 characters has been read or a zero record size has been encountered during the reading of a TECHNIQUE-B or TECHNIQUE-C file which utilizes the SIZE DEPENDING option. The program is DS-ED.
27	Not used.
28	While operating under Time Sharing, a SEEK has been issued for a Data Communications device. The program is DS-ED.
29	An irrecoverable parity error has occurred while reading a tape file which was opened REVERSED. The message will be typed once for each block which is in error unless a USE procedure has been specified. The USE procedure (if any) will be executed and control will be transferred to the statement following the READ statement.
30	Not used.
31	An attempt was made to READ from a file which is closed or was never opened. The program is DS-ED.
32	An attempt was made to WRITE to a file which is closed or was never opened. The program is DS-ED.
33	An attempt was made to SEEK on a file which is closed or was never opened. The program is DS-ED.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

<integer> value	Meaning
34	An attempt was made to WRITE BLOCK on an INPUT file. The program is DS-ED.
35	An attempt was made to WRITE BLOCK on a file opened REVERSED. The program is DS-ED.
36	Not used.
37	An attempt was made to WRITE BLOCK on a file which is closed or never opened. The program is DS-ED.
69	An attempt was made to write on disk at an address less than 100. The program will hang in a DO UNTIL FALSE loop. The program may be DS-ED by the operator.
71	The number of records within a string on a tape, used by a COBOL SORT program, was wrong (this was due to an incorrect Read or Write on that tape). Consequently, processing of the program was discontinued.
76	An error occurred within a string being written by a COBOL SORT Program; the number of records that should have been written did not equal the number written on the designated unit. Consequently, processing of the program was discontinued.
79	The number of records that should have been read from other tape units in the final merge pass of a SORT, being performed by a COBOL SORT Program, did not equal the number of records written onto the final output tape. However, after action was taken to type this message, the SORT closed the final output reel or executed the user's output routine, signaling End-Of-File. Consequently, the output tape may be used in spite of this error message. (The tape unit indicated in this message is meaningless.)
80	The total number of records entered as input to the SORT, being performed by a COBOL SORT Program, was not equal to the number of records produced as output from the SORT in the final merge pass. However, after action was taken to write this message, the SORT closed the final output file or executed the user's output routine signaling End-Of-File. Consequently, the output tape may be used in spite of this message. (The tape unit indicated in this message is meaningless.)
81	The amount of disk available is insufficient for a disk-only or ITD mode sort.
82	The number of records read from the input does not match the number written to the final output.
83	A disk file was passed as an output file which was not large enough to hold all of the sorted output data.

<integer> value	Meaning
84	Disk-only mode. The amount of disk specified is insufficient to do a disk-only sort.
85	ITD mode. The number of records read from a string on tape is not the same number written.
86	No records have been passed to either a COBOL or an ALGOL SORT Program.

<unit mnemonic> I/O INV ADDR

The occurrence of this message denotes that an invalid address occurred when data was to be transferred between an I/O channel and core memory. The MCP recognizes this error condition and rectifies the errors if possible. The primary purpose of this message is to draw attention to a condition which, if it occurred frequently, could denote a hardware failure.

<unit mnemonic> I/O MEM PAR

The occurrence of this message denotes that a memory parity error occurred during the transfer of data between an I/O channel and core memory. The MCP recognizes this error condition and rectifies the errors if possible. The primary purpose of this message is to draw attention to a condition which, if it occurred frequently, could denote a hardware failure.

LIB MAINT ERROR, <unit mnemonic> , <error condition>

This message indicates that an error has been detected while a library tape is being loaded or created.

-LIB MAINT ERROR, <unit mnemonic> , <file specifier> NOT LOADED

The occurrence of this message indicates that the specified file was not loaded due to an error. If the file was on disk before the load was started, it will be removed.

-LIB MAINT ERROR, <unit mnemonic> , DUMP ABORTED

This message indicates that a library dump has been aborted due to an error.

<file specifier> LIBRARY MAINTENANCE IGNORED

This message is typed if the MCP cannot perform the library maintenance operation specified on a control card.

<file specifier> LOADED

This message is typed after the MCP has performed an operation specified on a LOAD control card.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

LOG HALF FULL

This message is typed if the log file SYSTEM/LOG is half full as a warning to the operator, so that log information will not be lost because of a log wrap around.

LOGOUT/DISK AUTO SCHED

This message is typed if the MCP is required to automatically schedule the program LOGOUT DISK because of the fact that the file SYSTEM/LOG has been filled to its limit.

LOG WRAP AROUND

This message is typed if the MCP has to write on the beginning of the log file SYSTEM/LOG because of the fact that the log file has been filled and not reinitialized.

LP BACKUP ON <unit mnemonic>

This message is provided to notify the operator that a print backup tape is on-line. (No operator action is required unless it is desired to print the tape. If the tape is to be printed, a PB message must be entered.)

LP, PBT MT RQD <data file designator> <rdc> : <job specifier>

The occurrence of this message indicates that a program has need for a line printer or printer backup tape and neither is available. The situation denoted by this message will be remedied if a line printer, backup tape, or scratch tape becomes available. The nature of the condition can be altered through use of the OU message.

LP RQD <data file designator> <rdc> : <job specifier>

The occurrence of this message indicates that a program has need for a line printer and no such I/O device is currently available. The situation denoted by this message will be remedied when a line printer becomes available; however, the OU message may be used to alter the nature of the condition.

-MAXN ARGMNT EXP: <job specifier> <terminal reference>

This message is typed if the argument to the EXP intrinsic exceeds 158, which will cause the program to be terminated.

MCP INCLUDES <DATACOM> , <DCSPO> , <DCLOG> , <DFX> , <DUMP> , <DEBUGGING> , <BREAKOUT>

This is the message SPOed, conditional on the compile parameters set TRUE, for the WM message provides a method of determining the compiled version of the MARK VIII MCP being used.

-MEMORY PARITY <job specifier> , <terminal reference>

The occurrence of this message denotes that a hardware failure has occurred which precluded the further processing of the designated program. Consequently, processing of the program was discontinued.

MORE THAN 12000 CARDS IN <control card>

This message is typed when there are more than 12000 cards in a card deck which is being placed on the disk by LDCNTRL/DISK. This card deck is then completely removed from the disk.

MCP <version> . <release level> INCLUDES <MCP module list>

This message is typed in response to the keyboard input message WM. The message identifies the current MCP version and the patch level and a list of the options under which the MCP was compiled.

MT RQD <data file designator> <rdc> : <job specifier>

The occurrence of this message indicates that a program is in need of a scratch tape to use for a magnetic tape file.

-NEGTV ARGMNT LN <program specifier> <termination reference>

This message will be typed upon the occurrence of a negative argument being passed to the LN intrinsic.

-NEGTV ARGMNT SQRT <program specifier> <termination reference>

This message will be typed upon the occurrence of a negative argument being passed to the SQRT intrinsic.

NEW PBT ON <unit>

This message is printed when a new printer backup tape is opened.

NO BACKUP DISK

The occurrence of this message denotes that a data overlay operation was required, but no backup disk (i.e., overlay storage) was available. If the condition indicated by this message should occur, a HALT-LOAD operation is required.

NO FILE <data file designator> <rdc> : <job specifier>

The occurrence of this message denotes that a program has need for an input file which is apparently not available. If the subject file is labeled, the situation denoted by this message may be remedied by making the file available.

If the file is labeled, the IL message must be used. If the file is a COBOL optional file, an OF message may be entered. If a COBOL Program has read the final reel of a multi-reel unlabeled file, the FR message may be entered.

NO FILE <library tape name> / FILE000

This message occurs when an attempt is made to LOAD files from a library tape which is not available to the system.

NO FIL ON DISK <data file designator> : <job specifier>

The occurrence of this message denotes that a program has need for a file it expected to find on disk. If the file noted in this message is made available on the disk so that the subject program can continue processing, the <mix index> OK message must be entered; then the MCP will again search for the file to make it available to the program. If the file noted in the message cannot be made available, a DS message should be entered for the subject program.

<mix index> NO MEM

The occurrence of this message denotes that the MCP has made an attempt to obtain an area in core memory, but was unable to do so. After not obtaining the area, the MCP allows other processing, if any, to take place; and subsequently makes periodic attempts to obtain the desired area. If the area is ever obtained, the OK MEM message will be typed. The <mix index> in this message denotes the program for which the area was to be obtained; MIX = 0 denotes the MCP. When the NO MEM message appears, it may or may not be followed by an OK MEM message. The system operator is required to determine actions subsequent to the NO MEM message; a HALT-LOAD operation may be required.

<program-id> NOT EXECUTABLE CODE

The occurrence of this message indicates that the MCP has performed a consistency check on a program and the program was not of executable form.

NO SM STATIONS ON MIX = <mix index>

The mix SS message provides a means whereby a message can be sent to all "remote SPO" users of a particular mix, who have requested mix messages via the SM message. The NO SM STATIONS ON MIX is typed if no users have requested messages.

<file specifier> NOT IN DIRECTORY

This message is typed if a control card references a file which is not in the disk directory.

<file specifier> NOT LOADED (NOT ON TAPE)

This message is typed if a LOAD control card references a file which is not on the specified library tape.

<unit mnemonic> NOT READY

The occurrence of this message denotes that the MCP or an object program has attempted to perform an I/O operation on the designated unit and has found the unit NOT READY.

<unit mnemonic> NOT READY EU

The occurrence of this message denotes that the MCP or an object program has attempted to perform an I/O operation on the designated unit and has found the disk file electronic's unit not ready.

NO USER DISK

This message will occur if the MCP is requested to perform a library maintenance activity which requests an area on user disk and no such area is available. If the condition indicated by this message should occur, a HALT-LOAD operation is required.

NO USER DISK: <job specifier>

The occurrence of this message denotes that a program has attempted to obtain a file area on user disk, but an area of the required size is not available. If subsequent action is taken to make user disk available, the OK message must be entered to cause the MCP to again attempt to find the requested area. If no user disk is made available, a DS message should be entered for the program.

-NULL LIBRARY DEF

This message indicates that a DUMP function proved to be vacuous. In other words, the item or items to be DUMPed do not exist on the Disk. If, for example, the operator requests that files A /= be DUMPed to a library tape called DEF and there are no files for which the first name is A, then the message is typed.

NULL REMOTE/LOG

This message is typed the first time the remote log is accessed and is not present.

NULL PBxxxx

This message is typed if an attempt is made to print a printer backup file while the "PRNPBT/DISK" routine is not in the directory, a printer is not available, or the file does not exist.

<mix index> OK MEM

This message may occur after a NO MEM message. The occurrence of this message denotes that the condition indicated by the NO MEM message no longer exists.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

OPRTR ST-ED <job specifier>

The occurrence of this message means that the job has been suspended in response to an ST keyboard input message.

-OPRTR DS-ED <job specifier> , <termination reference>

This message is typed after the system operator causes processing of a program to be discontinued through use of a DS message.

<unit mnemonic> OUT <data file designator> <rdc> : <job specifier>

This message is typed when a program opens a card, tape, or line printer file for output, providing the necessary options have been set. The message will be typed for object program files if the TYPE OPN option is set. The message will be typed for compiler files if both the TYPE OPN and TYPE CMPLRFIL options are set.

<unit mnemonic> OUT PBTMCP BACKUP : <job specifier>

This message is typed when a scratch tape is initially selected and used for a printer backup tape, providing the necessary options have been set. The message will be typed when an object program places the first file on a printer backup tape if both the TYPE OPEN and TYPE CMPLFILE options are set.

PARITY ON <unit mnemonic>

The occurrence of this message means that the MCP has tried to read this tape and received an irrecoverable parity condition while reading the label information or scanning down a multi-file reel.

<unit mnemonic> PARITY, RW/L

The occurrence of this message indicates that the MCP has attempted to read the designated magnetic tape unit, but has received a parity error condition and has consequently made the unit inaccessible. The reason for the apparent parity condition might be that the tape unit has been set to the wrong density. If the subject unit is made ready again -- either by placing the unit in LOCAL and then in REMOTE or through use of the RY message -- the MCP will make another attempt to read the tape. Also, a PG message referencing the subject unit can be entered, and the tape will be purged and made accessible.

-PAR NO LABEL <file designator> : <job specifier> , <termination reference>

The occurrence of this message that there was an irrecoverable parity on the designated file and the object program did not specify any action for such a condition. Consequently, processing of the program was discontinued.

PBT MT RQD <data file designator> <rdc> : <job specifier>

The occurrence of this message indicates that a program is in need of a scratch tape to use for a printer backup file. The situation denoted in this message will be remedied when a scratch tape is made available. The nature of the condition can be altered through use of the OU message.

<unit mnemonic> PG-ED

This message is typed when a tape is purged either by a keyboard input message or a program.

<unit mnemonic> PRINT CHECK

This message is typed when a print check error has occurred during printing of a line on a line printer. This message is provided for the purpose of notifying the operator that the error has occurred; processing of the program using the line printer is continued as though the error had not occurred.

PP RQD <data file designator> <rdc> : <job specifier>

The occurrence of this message denotes that a program has need for a paper tape punch and no such I/O device is currently available.

<unit mnemonic> PUNCH CHECK

This message is typed when a punch check error has occurred during the punching of a card. This message is provided for the purpose of notifying the operator that the error has occurred; processing of the program using the card punch is continued as though the error had not occurred.

<unit mnemonic> READ CHECK

This message is typed when a read check occurs on a card reader. The operator must put the card through the card reader again. If the card is a badly worn card, it should be reproduced.

READ ERROR FOR control card information

The occurrence of this message denotes that a read error, probably irrecoverable parity, has occurred during the reading of a control deck for the disk. The control card which is printed out denotes the deck which will be deleted because of this error. The following decks will still be loaded.

REMOTE/LOG FULL

This message is typed when the log is full. Wrap-around will occur the next time the log is accessed.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

REMOTE LOG ON DISK

This message is SPOed when the REMOTE/LOG has been placed on disk and initialized. If the file with <file identification prefix> REMOTE and <file identification> LOG is not on disk, then 1000 segments are obtained for the file REMOTE/LOG and it is entered in the Disk Directory. The first 1000-ABRTLNGTH segments are reserved for log-entries; the record capacity in logical records of this area equals $6^{\times}$ (1000-ABRTLNGTH). The remaining segments are reserved for information pertinent to remote terminals currently attached to programs for abort logging if necessary. (An entry is made in this section of the file for each remote terminal attached to a job.) The maximum number of such entries is $3^{\times}$ (ABRTLNGTH-1).

<file specifier> REMOVED

This message is typed after the MCP has performed an operation specified on a REMOVE control card.

-RER NO LABEL <file designator> : <job specifier> , <termination reference>

The occurrence of this message denotes that there was an R-format error on the designated input file and the object program did not specify any action for such a condition. Consequently, processing of the program was discontinued.

-RER NO LABEL <prog. id.> <termination reference>

This message is printed if an R-editing phrase error is detected on an array row read statement.

-H/L MARK DCMCP <Roman numeral> . <integer> MODS RRRRRRRR-

This message is typed immediately following a HALT-LOAD operation. The Roman numeral identifies the level of the MCP and the integer indicates the number of changes to the basic level. An @ appearing in the string of R's indicates a memory module that is not ready.

<unit mnemonic> RW/L

The occurrence of this message denotes that an operation has been performed to rewind the tape on the designated unit and to make the unit inaccessible. The unit may be made accessible again by placing it in LOCAL and then REMOTE, or through use of the RY message.

<unit mnemonic> RW/L <library tape name> (LIBRARY DUMP)

This message occurs after a library tape has been made through use of the DUMP card facility. The designated unit is the location of the newly created library tape, and the unit has been made inaccessible again by placing it in LOCAL and then in REMOTE, or through use of the RY message.

-SELECT ERROR <file designator> : <job specifier> , <termination reference>

The occurrence of this message denotes that an object program performed an invalid operation on the designated file; e.g., rewinding a card reader. Consequently, processing of the program was discontinued.

SLATE OVRFLW

The occurrence of this message denotes that too many entries have been made in the SLATE, a table used by the control section of the MCP. If the condition indicated by this message should occur, a HALT-LOAD operation is required.

STA <integer> / <integer> MARKED NON-SPO

This is an output SPO message. The DC MCP has a means for "remembering" the locations of remote stations which have SPO capabilities. It is sometimes desired, however, to designate that certain stations should no longer be identified as "remote SPO's" (e.g., when an adapter which does not facilitate SPO facilities replaces a TWX adapter). In order that stations may be marked as "non-SPO stations", the RR keyboard input message is provided. If a message of the form RR <integer> / <integer> is entered -- where the first <integer> specifies a terminal unit number and the second a buffer number -- the MCP will remove that station's identify as an "SPO station". Also, if the station is logged-in, it will log it out.

-STACK OVRFLW <job specifier> , <termination reference>

The occurrence of this message denotes that the operations performed by an object program have caused its stack to overflow its limit. Consequently, processing of the program has been discontinued.

STATION <terminal unit> / <buffer> DISCONNECT AT <time>

This message is typed if a remote station becomes detached from the system without properly logging out.

STATION <terminal unit> / <buffer> LOGGED IN AT <time> BY <user code>

This message is typed when a user logs in from a remote station. If the file REMOTE/USERS is not present, <user code> will be <empty>.

STATION <terminal unit> / <buffer> LOGGED OUT AT <time>

This message is typed when a user logs out from a remote station.

<compiler name> / <program identifier> = <mix index> SYNTAX ERR

This message is typed when a compiler reaches End-of-Job and the program being compiled contains syntax errors, providing the TYPE EOJ option was set.

PRINTED IN U.S. AMERICA-REVISED 8/70 1031986

<unit mnemonic> TAPE MK, RW/L

The occurrence of this message indicates that the MCP has attempted to read the designated magnetic tape unit, has found the first word of information to be a tape mark, and has consequently made the unit inaccessible. The reason for the apparent tape mark condition may be that the tape unit has been set to the wrong density. If the subject unit is made ready again -- either by placing the unit in LOCAL and then in REMOTE, or through use of the RY message -- the MCP will again attempt to read the tape. Also, a PG message referencing the subject unit can be entered, and the tape will be purged and made accessible.

TR PLEASE

This message is typed at HALT-LOAD time if the TYPE TIME option is set. The system operator is required to enter a TR message before processing can continue.

UNEXP IO ERR

The occurrence of this message denotes that the MCP encountered an unexplained I/O error that could not be directly associated with a particular program. If this error should occur, a HALT-LOAD operation is required.

-UNEXP I-O ERROR, <unit mnemonic>, LIB MAINT ABORTED

The occurrence of this message indicates that a library load or dump has been aborted due to an unexpected I-O error. If this occurs during a load operation, the file currently being loaded will be removed from the directory.

UNEXP I-O ERROR ON <unit> : RESULT = <error field> ; MASK = <mask> .

Where:

<unit> ::= The Unit Mnemonic (CRA, MTC, DKA, etc.).
<error field> ::= Result descriptor error field (26:7).
<mask> ::= Mask specified by the calling procedure. A bit on in the mask indicates a condition to be handled by the calling procedure.

The message indicates that the MCP has performed an I/O operation which caused an unexpected I/O error. If the I/O operation is directly related to a particular object program, processing of that program is discontinued. Also, the MCP types the Error Field, the Unit Mnemonic, and the Error Mask Field as additional information.

<file name> UNOPENED

This message states that the I/O area was referenced while the file was not open.

<unit mnemonic> WRITE LOCK

The occurrence of this message denotes that a program has attempted to write on a magnetic tape with no write ring or on a disk or drum, which has been locked out through use of hardware lockout switches. Consequently, processing of the program using the unit has been discontinued.

<unit mnemonic> WR PARITY

The occurrence of this message denotes that an irrecoverable write parity has occurred on the designated unit. Consequently, processing of the program using the unit has been discontinued.

ZIP ERROR - IGNORED

This message is typed if a program performs a generalized ZIP statement, but provides control information containing an error. Occurrence of this message signifies that the error was present and that all control information following and including the error was ignored.

-ZERO ARGMNT LN <program specifier> <termination reference>

This message will be typed upon the occurrence of an argument of zero being passed to the LN intrinsic.

KEYBOARD INPUT MESSAGES

Operator messages are defined as messages with a free-field format which the operator can supply the MCP via the B 5500 console keyboard. In keeping with the concept of permitting the system to perform the control functions, the operator messages are primarily restricted to those actions that will facilitate processing. The messages are not intended to provide detailed information about individual programs; e.g., the settings for specific registers or the contents of designated memory locations.

To enter information from the keyboard, the operator must first depress the INPUT REQUEST key. The READY indicator on the supervisory printer is turned on. At this time, the operator can enter his message. When he has finished keying in the message, he depresses the END OF INPUT key. This key causes a group mark to be inserted immediately after the last character entered and signals the MCP.

If the operator attempts to introduce a message that is not acceptable, the MCP will not act upon it, except to notify the operator that he has keyed in an invalid entry.

The following list presents the allowable operator input messages.

THE AX MESSAGE

The AX message allows the console operator to communicate with the object program through the SPO in response to an ACCEPT message.

All characters following the AX including blanks will be available to the object program. Following the last character typed will be a group mark.

The AX message has the following format:

<mix index> AX<message data>

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

THE BK MESSAGE

Performs the equivalent of the break key function for a SPO console or the SPO.

The BK message has the following format:

BK

Example:

BK

THE BS MESSAGE

Sets the station indicated by <TU> / <BUFF> as a SPO console. (See description of SPO consoles.)

The BS message has the following format:

BS <tu> / <buff>

Example:

BS 1/8

THE BS SPO MESSAGE

Causes SPO output to be printed on the SPO.

The BS SPO message has the following format:

BS SPO

Example:

BS SPO

THE CC MESSAGE -- ? MESSAGE

The CC message allows the system operator to supply control information to the MCP via the console typewriter. The information following the letters CC in the CC message are recognized in the same fashion as the information following the character ? on control cards and program-parameter cards.

The character ? can be used in lieu of the characters CC in the CC message, if desired.

When a CC message is entered and the END OF INPUT switch is pressed, the typewriter will become READY again unless the CC message contained END card information. Consequently, the last CC message must always be an END card message.

The term <control information> used below is defined as any information defined valid for use on control cards or program-parameter cards.

The CC message may have either of the two following formats:

CC <control information>

or

? <control information>

Examples:

```
CC EXECUTE C/P; END
CC EXECUTE C/P
CC END

? COMPILE "00180" BY IRP WITH ALGOL
? ALGOL FILE CARD = IRACARD
? END
? COMPILE A/B; ALGOL FILE CARD = "0XXXXXX"; END.
```

THE CD MESSAGE

The CD message causes the MCP to type the name and first card image of each pseudo card deck that was placed on the disk by the LDCNTRL/ DISK Program. If there are no pseudo card decks on the disk, the following will be typed:

NO DECKS ON DISK

The CD message has the following format:

CD

THE CI MESSAGE

The CI message causes the MCP to forget the current intrinsic file. If, at a time when an intrinsic file is already on disk, a new intrinsic file has been created, and the operator wishes to remove the old file and use the new one, the CI (for Change Intrinsics') message may be used. If the designated file is found, the MCP will wait until the only jobs being processed are LDCNTRL/DISK and PRNPBT/DISK (or the mix is null), and then perform the change.

The CI message has the following format:

CI <file identification prefix> <separator> <file identification>

Example:

CI INT/DISK

THE CL MESSAGE

The CL message will discontinue the job using the specified unit.

The CL message has the following format:

CL <unit mnemonic>

Example:

CL MTA

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

THE CT MESSAGE

The CT message is used to change the time limits for a job. For either I/O or processor time limits, if the input is a non-zero integer, then the time limit will be changed to these new values.

The CT message has the following format:

<mix index> CT <processor part> <I/O part>

THE CU MESSAGE

The CU message allows the console operator to determine the core usage for a single program or all programs in the mix.

The CU message has the format:

<mix index> CU

or

CU

THE DS MESSAGE

The DS message allows the system operator to cause a program to be terminated.

There are two forms of the DS message. One form of the message requires that the program to be terminated be identified through use of a <mix index>* term; the other message requires that the program be identified through use of a <program specifier>.

If more than one program in a MIX have the same <program name> and a message using a <program specifier> is entered, the MCP will arbitrarily terminate the program -- with the name specified -- that has the lowest <mix index>. Consequently, if a situation such as noted should occur, the DS message which identifies the program through use of the <mix index term> should be used.

The DS message may have either of the two following formats:

<mix index> DS

or

DS <program specifier>

Examples:

2 DS
DS ALGOL/"00180"

* The term <mix index> is an <integer> that represents the MIX index that the MCP has assigned to a particular program in the MIX; i.e., a program that is currently in process.

THE DT MESSAGE

The DT message allows the system operator to change the value of the current date word used by the MCP.

The DT message requires the use of three <integer>s, the first two of which must be followed by the character /. The first <integer> is recognized as the number of the month of the year, the second <integer> is recognized as the day of the month, and the third <integer> is recognized as the last two <digit>s of the year.

The DT message must have the following format:

DT <integer> / <integer> / <integer>

Example:

DT 6/1/70

THE ED MESSAGE

The ED message can be used to eliminate a pseudo card deck which is contained in a pseudo card reader if the reader is not in use.

The ED message may have one of the following formats:

ED CDA
ED CDB
ED CDC
ED CDD

THE EI MESSAGE

The EI message responds with the message EIO and performs no useful function.

THE ES MESSAGE

This message terminates a program which is still in the schedule. This cannot be done with a DS message because the program is in the schedule and not in the mix. The program may be eliminated from the schedule by typing in ES <job specifier> (meaning to eliminate from schedule). This will cause the program to be loaded into the mix and DS-ED to be performed before any of its statements are executed.

THE EX MESSAGE

The EX message types out all expired disk file names.

Examples:

EX ALGOL/=
EX =/DISK
EX

THE FM MESSAGE

The FM message must be entered in response to a # FM RQD message. The <mix index> in the message must agree with the <mix index> in the # FM RQD message, and the <unit mnemonic> must designate the unit to be used for the subject file.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

The FM message has the following format:

<mix index> FM <unit mnemonic>

Example:

1 FM LPB

THE FR MESSAGE

The FR message allows the system operator to specify that the input reel, the reading of which was just completed, was the final reel of an unlabeled file.

The FR message has the following format:

<mix index> FR

Example:

3 FR

THE IL MESSAGE

The IL message is used in response to a no-file message and allows the system operator to designate the unit on which a particular input file is located. The unit designated in the IL message may denote the location of a non-standard file (a file with no standard B 5500 label) or a standard file (a labeled file). In either case, the file on the unit designated in the IL message will be assumed to be the file required in the related no-file message.

A <mix index> term must be used with the IL message since, during multiprocessing, more than one no-file message may be in effect at the same time.

The IL message must have the following format:

<mix index> IL <unit mnemonic>

Example:

1 IL MTF

THE IN MESSAGE

The IN message allows the system operator to insert an <unsigned integer> into the Program Reference Table (PRT) of the program specified by the <mix index> at the relative location specified by the octal <index> unless the specified PRT cell contains a descriptor, or the <index> is less than 25 <octal> or out of the PRT bound.

The IN message has the following format:

<mix index> IN <index> = <unsigned integer>

Example:

2 IN 32 = 563

THE LD MESSAGE

The LD message causes the LDCNTRL/DISK Program to be called out for execution. The LDCNTRL/DISK Program then searches for a tape or card file with the <multiple file identification>

CONTROL

and the <file identification>

DECK

Then, if the message entered was

LD DK,

the file CONTROL/DECK is placed on disk in such a fashion that the MCP can read the file as a pseudo card deck. If the message entered was

LD MT,

the file CONTROL/DECK is placed on a magnetic tape.

The LD message may have either of the following formats:

LD DK

or

LD MT

Examples:

LD DK
LD MT

THE LN MESSAGE

The LN message causes the library program with the <program identifier>

LOGOUT

and the <program identifier suffix>

DISK

to be scheduled for execution.

The LN message has the following format:

LN

THE LR MESSAGE

The LR message causes the library program with the <program identifier>

LOGOUTR

PRINTED IN U.S. AMERICA REVISED 8/71

1031986

and the <program identifier suffix>

DISK

to be scheduled for execution.

The LR message has the following format:

LR

Example:

LR

THE MX MESSAGE

The MX message allows the system operator to request that the MCP type a list of <program specifier>s denoting the programs in the MIX; the priority and <mix index> for each program is also listed.* Specifically, each item in the list typed by the MCP in response to the MX message has the following format:

<priority> : <program specifier> = <mix index>

If there is nothing in the MIX, the following message will be typed:

NULL MIX

The MX message must have the following format:

MX

THE OF MESSAGE

The OF message allows the system operator to specify that a file requested for a COBOL Program was optional, so that the specified program can proceed without it.

The OF message has the following format:

<mix index> OF

Example:

1 OF

THE OK MESSAGE

The OK message causes the MCP to resume processing of a program which has been temporarily suspended because of the condition designated by the # DUP LIBRARY

* It should be noted that the maximum number of programs allowed in the MIX is determined by two parameters which are DEFINED in the MCP. The two parameters are MIXMAX and JOBNUMAX, where MIXMAX may be DEFINED as an integer from 1 through 29 and JOBNUMAX must be DEFINED as an even integer with a value of 20 x MIXMAX + 30.

message, the NO USER DISK message, the NO FILE ON DISK, or the #OPRTR ST-ED message.

The OK message has the following format:

<mix index> OK

Example:

1 OK

THE OL MESSAGE

The OL message allows the system operator to request that the MCP type information pertaining to labels of files on I/O units.

The OL message has many formats. One format specifies that a specific <unit mnemonic> may be entered. The other formats require two-letter codes which specify a type of I/O unit. The codes and the I/O units they represent are as follows:

Code	I/O Unit
CD	Pseudo Card Reader
CP	Card Punch
CR	Card Reader
LP	Line Printer
MT	Magnetic Tape
PP	Paper Tape Punch
PR	Paper Tape Reader

If an OL message specifying a specific <unit mnemonic> is entered, the response message will have one of the following formats, whichever is relevant.

<unit mnemonic> IN USE BY <program specifier> : <multiple file identification> <file identification> <rdc>

<unit mnemonic> LABELED <multiple file identification> <file identification> <rdc>

<unit mnemonic> NOT READY
<unit mnemonic> SCRATCH
<unit mnemonic> UNLABELED

If an OL message specifying a type of I/O unit is entered, and if a unit of the specified type is in use and/or labeled, the response message will have one of the following formats, whichever is relevant.

<unit mnemonic> IN USE BY <program specifier> : <multiple file identification> <file identification> <rdc>

<unit mnemonic> LABELED <multiple file identification> <file identification> <rdc>

<unit mnemonic> UNLABELED

If an OL message specifying a type of I/O unit is entered, and no unit of that type is in use and/or labeled, the following message will be typed:

NULL <unit mnemonic> TABLE

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

The OL message may have one of the following formats:

OL <unit mnemonic>

or OL CD or OL CP or OL CR or OL LP or OL MT or OL PP or OL PR.

Examples:

OL MTA
OL CR
OL MT
OL MTX (where X calls for a list of all scratch tapes).

THE OT MESSAGE

The OT message allows the system operator to request the MCP to type out the value of a cell in a program's Program Reference Table (PRT). The program is specified by the <mix index> and the cell by the octal <index>. The MCP message typed will have the following format:

<job specifier> : R+ <index> = <PRT data>

The value of <PRT data> will be expressed as an octal number for a descriptor, or an integer of up to eight digits for an operand.

The OT message has the following format:

<mix index> OT <index>

Example:

2 OT 32

THE OU MESSAGE

The OU message allows the system operator to designate the output media option for a line printer file if a # LP RQD, a # LP PBT MT RQD, or a # PBT MT RQD message has been typed which references the job that uses the file.

The OU LP form of this message specifies that the subject line printer file must be produced as output on a line printer.

The OU MT form of this message specifies that the subject line printer file must be produced as output on a printer backup tape.

The OU DK form of this message specifies that the subject line printer file must be produced as output on printer backup disk.

The OU form of this message specifies that the subject line printer file may be produced as output either on a line printer or a printer backup tape. The OU message may have any one of the following formats:

<mix index> OU LP or <mix index> OU MT or <mix index> OU or
<mix index> OU DK

Examples:

2 OU LP
1 OU
4 OU

THE PB MESSAGE

The PB message allows the system operator to specify that a printer backup file on a particular unit is to be printed. If a specified tape is not a printer backup tape, the following message will be typed:

NOT PRINTER BACKUP TAPE

The term <PBD number> may be up to four digits and is the nnnn part of the PBD file name.

The PB message has the following formats:

PB <unit mnemonic>
PB <PBD number>

Example:

PB MTN

THE PD MESSAGE

The PD message allows the system operator to request that the MCP type information pertaining to what files are listed in the disk directory. The formats of the PD message are shown below. The action caused by the PD message depends upon the format of the message. Specifically, the actions caused by the PD message are as follows.

If a message of the form

PD = / =

is entered, a list containing a <file specifier> for each file in the disk directory is typed.

If a message of the form

PD <file specifier>

is entered and the file designated in the message is in the disk directory, the <file specifier> for the file will be typed. If the file designated in the message is not in the disk directory, the message

NULL PD <file specifier>

will be typed.

If a message of the form

= / <file identification>

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

or

= / <program identification suffix>

is entered, a list of all files in the disk directory which have the designated <file identification> or <program identification suffix>, if any, will be typed. If no such files are in the disk directory, a message of the form

NULL PD <file identification prefix> / =
or NULL PD <file identification prefix>
or NULL PD <program identification> / =
or NULL PD <program identification>

will be typed.

In total, the PD message may have any one of the following formats:

PD = / =	PD <file identification prefix> / =
PD <file specifier>	PD <file identification prefix>
PD = / <file identification>	PD <program identification> / =
PD = / <program identification suffix>	PD <program identification>

Examples:

PD = / =
PD ALGOL/DISK
PD = / PARTS
PD = / DISK
PD PERSNEL / =
PD PERSNEL
PD ALGOL / =
PD ALGOL

THE PG MESSAGE

The PG message allows the system operator to purge a magnetic tape on a unit that is READY, in WRITE status, and not in use.

The PG message has the following format:

PG <unit mnemonic>

Example:

PG MTK

THE PR MESSAGE

The PR message provides a means whereby the system operator can specify the priority to be assigned a program currently in the MIX. The priority to be assigned is specified by the term <priority>; the program to which the priority is to be assigned is specified by the <mix index>. The term <priority> must be an <integer>.

The PR message has the following format:

PR <mix index> = <priority>

Example:

PR 1 = 7

THE PS MESSAGE

The PS message can be used to alter the priority of a program in the schedule. The priority to be assigned is specified by the term <priority>; the program in the schedule to which the priority is to be assigned is specified by the <schedule index>. Both terms are integers.

The PS message has the following format:

<schedule index> PS = <priority>

Example:

5PS = 3

THE QT MESSAGE

The QT message is used to cause PRNPBT/DISK to skip a printer backup file. In response to this message, the following actions will be taken:

1. Printing of the current file is discontinued.
2. Printing resumes with the first record of the succeeding file.

The QT message may have either of two formats:

<mix index> QT

or

QT <unit mnemonic>

Examples:

1 QT
QT MTA

THE QV MESSAGE

The Q-second timeout facility provides a means for causing the MCP to clear read-ready conditions from the terminal unit buffers of remote stations which have SPO capabilities in cases where an object program (to which the station is assigned) does not read the input within a given number of seconds; i.e., within "Q" seconds. The value of Q may be specified through use of a QV keyboard input message of the form:

QV = <integer>

In response to this message, the MCP will set Q to the value specified and output an SPO message of the form:

REMOTE SPO Q VALUE = <integer> SECS

Also, if a keyboard message of the form

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

QV

is entered, the MCP will respond with a message, as shown above, which specifies the current value of Q.

THE RD MESSAGE

The RD message may be used to remove pseudo card decks from disk which were placed on disk by the LDCNTRL/DISK Program.

Pseudo card decks are identified by names having the following format:

<integer>;

and the term <pseudo card deck list> is defined as:

<integer> | = or<pseudo card deck list>

or

<pseudo card deck list>

Examples:

RD #0072
RD #0072, #6328
RD =

THE RM MESSAGE

The RM message can be used in response to a # DUP LIBRARY message. The RM message causes the file on disk (with a name identical to the file created by the program specified in the # DUP LIBRARY message) to be removed, and then causes the subject program to resume processing.

The RM message has the following format:

<mix index> RM

Example:

1 RM

THE RN MESSAGE

The RN message is used to specify the number of pseudo card readers to be used. In total, there are four pseudo card readers. At HALT-LOAD time, the number of pseudo card readers specified to be used is zero.

An RN message may be entered at any time. If an RN message specifies that more pseudo card readers are to be used than are currently being used, the MCP will search for pseudo card decks on disk and make use of as many of the specified pseudo card readers as possible. If an RN message specifies that fewer pseudo card readers are to be used than are currently being used, a sufficient number of the pseudo readers will be "turned off" as soon as the readers complete handling of the pseudo card deck in process, if any.

If no <digit> is entered, the digit 1 is assumed.

The RN message has one of the following formats:

RN

or

RN <digit>

Examples:

RN
RN 0
RN 1
RN 2
RN 3
RN 4

THE RR MESSAGE

The DC MCP has a means for "remembering" the locations of remote stations which have SPO capabilities. It is sometimes desired, however, to designate that certain stations should no longer be identified as "remote SPO's"; e.g., when an adapter which does not facilitate SPO facilities replaces a TWX adapter. In order that stations may be marked as "non-SPO stations", the RR (Remove Remote) keyboard input message is being provided. If a message of the form

RR <integer> / <integer>

is entered (where the first <integer> specifies a terminal unit number and the second a buffer number) the MCP will remove that station's identity as an "SPO station". Also, if the station is logged-in, it will log it out.

THE RW MESSAGE

The RW message allows the system operator to cause a rewind-and-lock action to be performed on a magnetic tape file that is not in use.

The RW message has the following format:

RW <unit mnemonic>

Example:

RW MTE

THE RY MESSAGE

The RY message allows the system operator to cause, by entering a keyboard message, an effect analogous to the effect caused by placing a magnetic tape unit in LOCAL and then REMOTE. That is, if the designated unit is not in use and in REMOTE, the MCP will attempt to read a file label.

The RY message causes locked files to be made accessible and causes label cards (or DATA cards), which have been read but not referenced, to be ignored.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

The RY message has the following format:

RY <unit mnemonic>

Examples:

RY MTC
RY CRA

THE SC MESSAGE

The SC message types the SPO consoles.

The SC message has the following format:

SC

Example:

SC

THE SO, RO, AND TO OPTION MESSAGES

The MCP provides a number of features that are optional. That is, if a particular option is set, the MCP will use the respective feature; if the option is reset (i.e., not set), the feature will not be used. See page 3-11 for an explanation of options.

The SO message allows the system operator to set options.

The RO message allows the system operator to reset options.

The TO message allows the system operator to request that the MCP type a message which lists the options and their settings.

Each optional feature provided by the MCP may be referenced either mnemonically through use of an <option mnemonic> or numerically through use of an <option numeric code>.

An <option mnemonic> is defined as one of the following:

USE DRA	TYPE DISCONDC
USE DRB	TYPE CMPLFILE
TYPE BOJ	TYPE CLOSE
TYPE EOJ	USE ERRORMSG
TYPE OPEN	USE RET
USE TERMNATE	USE LIBMSG
TYPE DATE	USE SCHEDMSG
TYPE TIME	USE SECMSG
USE ONEBREAK	USE DSKTOG
USE AUTOPRNT	USE RELTOG
USE CLEARWRS	USE PBDREL

An <option numeric code> is defined as:

USE OPTN <integer>

where the <integer> used specifies the option.

The SO message has either of the following formats:

SO <option mnemonic>
or
SO <option numeric code>

Examples:

SO TYPE BOJ
SO USE OPTN 45

The RO message has either of the following formats:

RO <option mnemonic>
or
RO <option numeric code>

Examples:

RO USE TERMNATE
RO USE OPTN 42

The TO message has the following format:

TO

THE SF MESSAGE

The Multiprocessing Factor can be changed by typing in SF <decimal number> (meaning to Set-the-Factor). The <decimal number> is defined as in ALGOL, with the restriction that <unsigned integer>s are at the most two digits long:

<decimal number> ::= <unsigned integer> | <decimal fraction>
<unsigned integer> <decimal fraction>

<decimal fraction> ::= . <unsigned integer>
<unsigned integer> ::= <digit> | <digit> <digit>

THE SS ALL MESSAGE

The SS ALL message provides a means whereby a message can be sent to all "remote SPO" users on the system.

The SS ALL message has the following format:

SS ALL: any characters excluding those having control significance

Example:

SS ALL: P.M. STARTS IN 30 MINS.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

THE <mix index> SS ALL MESSAGE

The <mix index> SS ALL message provides a means whereby a message can be sent to "remote SPO" users of a particular program, regardless of whether they have requested messages via the SM message or not.

If the given mix has no users, the message

NO STATIONS ON MIX = <mix index>

will be returned.

The mix SS ALL message has the following format:

<mix index> SS ALL: any characters aside from those having control significance

Example:

2 SS ALL: YOU MUST BE OFF BY 0900;
5 MIN. TO GO.

THE SS MESSAGE

The SS message may be used at the central SPO or, if preceded by a question mark, on a remote station with SPO capabilities. It directs a message to a remote station which has SPO capabilities, or to the SPO. If the station addressed is not recognized to have SPO capabilities or is Not Ready, an INV STN message is returned. The message, as provided at the addressed station, has a prefix which includes the address of the originator.

The SS message has the following format:

SS <remote station address> : <remote station message>

or

SS SPO : <remote station message>

Examples:

SS 1/O : ARE YOU THERE
? SS SPO : I NEED A SCRATCH TAPE

THE <mix index> SS MESSAGE

The <mix index> SS message provides a means whereby a message can be sent to all "remote SPO" users of a particular mix, who have requested mix messages via the SM message.

If no users have requested messages, the message

NO SM STATIONS ON MIX = <mix index>

will be returned.

The mix SS message has the following format:

<mix index> SS: {any characters aside from those having control significance}

Example:

3 SS: THE TAPE ON MTA IS ALMOST FULL.

THE ST MESSAGE

The ST message allows the system operator to suspend the program referenced by the <mix index> as soon as that program becomes ready to be returned to normal state by the MCP. To resume processing of the program, the operator must use the OK message.

The ST message has the following format:

<mix index> ST

Example:

1 ST

THE SV MESSAGE

The SV message may be used to cause a peripheral unit to be made inaccessible until a HALT-LOAD operation occurs or until an RY message referencing the inaccessible unit is entered. If, when the SV message is entered and the specified unit is not in use, the message

<unit mnemonic> SAVED

will be typed. If a unit is in use when an SV message referencing it is entered, the message

<unit mnemonic> TO BE SAVED

will be typed, and the unit will become inaccessible as soon as it is no longer in use. Until an RY message referencing the unit is entered or a HALT-LOAD occurs, the saved unit will not appear NOT READY.

The SV message has the following format:

SV <unit mnemonic> TO BE SAVED

Examples:

SV LPA
SV MTT
SV CRB

THE TF MESSAGE

The Factor can be interrogated by typing in TF (meaning to Type Out the Factor).

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

THE TI MESSAGE

The TI message causes the MCP to type out the amount of processor time that the subject program has used up at the time the TI message was entered. The time is provided as one to three <integer>s separated by <space>s.

For example:

1
or
2 49
or
1 48 7.

The right-most <integer> specifies seconds, the second-from-right integer specifies minutes, and the third-from-right integer specifies hours.

The TI message has the following format:

<mix index> TI.

Example:

3 TI

THE TL MESSAGE

The TL message provides a means whereby the MCP types the processor and I/O time limits for a designated job.

The TL message format is:

<mix index> TL

THE TR MESSAGE

The TR message allows the system operator to change the value of the time word used by the MCP.

The time, specified by the <integer> in the TR message, is designated according to a 24-hour clock; i.e., military time.

The TR message has the following format:

TR <integer>

Example:

TR 0800

THE TS MESSAGE

The TS message makes it possible to determine the programs in the schedule. The MCP will type out the names of each job in the schedule, together with the amount

of core space needed by the program and the amount of time the program has been in the schedule.

The format of the TS message is:

<priority> <job specifier> = <schedule #> IN FOR <elapsed time in schedule> NEEDS <core requirement>

THE UL MESSAGE

The UL message is used in response to a no file message, and allows the system operator to designate the unit on which a particular unlabeled file is located. The unit designated in the UL message may denote the location of a standard file (a file on which the first record is a standard B 5500 label) or a non-standard file (a file with no standard label). However, in either case all records on the file including the standard label, if any, will be recognized as data records. This message differs from the IL message in that, when the IL message is used in reference to a standard file, a standard label will not be recognized as a data record.

A <mix index> term must be used with the UL message since, during multiprocessing, more than one no-file message may be in effect at the same time.

The UL message has the following format:

<mix index> UL <unit mnemonic>

Example:

1 UL MTT

THE US SPO MESSAGE

The US SPO message inhibits SPO output to the SPO.

The US SPO message has the following format:

US SPO

THE <mix index> WA MESSAGE

The <mix index> WA message provides a means for determining that stations are assigned to a particular program.

If any stations are assigned to the given mix, the MCP will return a message of the form:

<integer> / <integer> ASSIGNED TO <program specifier>

If no stations are assigned to the given mix, the MCP will return the <mix index> WA message preceded by the word NULL.

The <mix index> WA message has the following format:

<mix index> WA

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

Example:

4 WA

THE WD MESSAGE

The WD message causes the MCP to type the date currently being used by the system. The date is given in the MM/DD/YY format.

The WD message has the following format:

WD

THE WM MESSAGE

The WM message allows the system operator to request the current version and release level of the MCP. A list of the module options under which the current MCP was compiled is also typed.

The WM message has the following format:

WM

Example:

WM

Response:

MCP VIII.78 INCLUDES DATACOM, DCSPO, DEBUGGING

THE WP MESSAGE

The WP message provides a means for determining what programs are assigned to what remote stations. If the WP message is followed by TU/BUF (where TU and BUF are each one or two digit numbers), the MCP will return a message specifying what programs are assigned to the specified stations, if any. If WP alone is entered, the MCP will return a complete list specifying what programs to what stations.

The message used to specify what programs are attached to what stations is as follows:

<integer> / <integer> ASSIGNED TO <Program specifier>

If no positive response can be provided for a WP message, the message will be returned preceded by the word NULL.

The WP message may have either of the following formats:

WP <integer> / <integer>

or

WP

Examples:

WP 2/6
WP

THE WT MESSAGE

The WT message causes the MCP to type out the time of day currently recognized by the system. The time is given according to a 24-hour clock.

The WT message has the following format:

WT

THE WU MESSAGE

The WU message provides a means for determining the user identifications of remote SPO users. If the WU message is preceded by a mix index, the MCP will identify the users of that mix, if any. If the WU is followed by TU/BUF (where TU and BUF are each one or two digit numbers), the MCP will identify the user of the given section, if any. If the WU is used alone, the MCP will identify all users of remote SPO stations, if any. The message used to identify the user of a remote station is as follows:

<integer> / <integer> USED BY <user code>

If no users are referenced by a WU message, the message will be returned preceded by the word NULL.

The WU message may have one of the following formats:

<mix index> WU

or

WU <integer> / <integer>

or

WU

Examples:

3WU
WU 1/4
WU

THE WY MESSAGE

The WY message allows the system operator to request that the MCP provide information as to why a program has been temporarily suspended, providing that the program has been temporarily suspended because of a reason previously designated in a system message which: (1) was preceded by the character # and (2) contained a <job specifier>; e.g., a program which was suspended because of the condition denoted by a previous # NO FILE message.

In response to the WY message, the MCP does the following: (1) lists the two-letter codes for all keyboard input messages which could be entered to eliminate the condition that caused the program to be temporarily suspended, and (2) retypes the #

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

message that was previously typed to inform the system operator of the condition that caused the program to be suspended.

The WY message has the following format:

<mix index> WY

Example:

4 WY

THE XI MESSAGE

The XI message (for "exchange intrinsics") causes the names of the files to be "swapped" when an installation wishes to interchange two copies of intrinsic files subject to the constraints mentioned in regard to the CI message.

The XI message has the following format:

XI <file identification prefix> <separator> <file identification>

Example:

XI NEW/INT

THE XS MESSAGE

The XS message causes a program which is in the schedule to be loaded in spite of the fact that the MCP does not think the program will run efficiently with the jobs already in the mix. This is done by typing in the new message XS <job specifier>, meaning to execute from schedule.

THE XT MESSAGE

The XT message is used to extend the time limits for a job. If <processor part> is <integer>, then the processor time limit will be extended by <integer> minutes. If either the processor of I/O time limits were extended, a message will be printed to notify the operator of this. In any event, the processor and I/O limits will be typed out.

The XT message format is:

<mix index> XT <processor part> <I/O part> <processor part> ::= <empty> | <integer> | * <I/O part> ::= <empty> | , <empty> | , <integer> | , *

THE REMOTE SPO STATION FACILITY

GENERAL

Remote stations are equipped with many of the capabilities of the supervisory printer. Most keyboard input messages can be utilized by the remote user and some system messages, such as NO FILE messages, can be printed on remote stations.

In order for a remote station to make use of SPO facilities, it must demonstrate that it is a typewriter station or TWX. One means whereby the MCP recognizes such a station is due to having received a WRU signal from that station. The WRU signal is automatically generated by a TWX after a remote operator successfully dials the computer. The WRU signal is also generated when the WRU key on a TWX or typewriter is pressed together with the control key. The second way in which a station is recognized as having SPO capabilities is if a log-in (LI) message is entered.

In order for a keyboard input message to be entered from a remote station, the operator is required to add a question mark as a prefix to the message to denote that it is directed to the MCP. Messages without a question mark prefix are assumed to be directed to an object program to which that station is attached.

It should be recognized that certain keyboard input messages are not allowed to be entered from remote typewriters. If such a message is entered, an INV KBD message will be returned.

When control cards are entered from remote stations, the keyboard input message may start with two question marks or a question mark followed by CC; i.e., a control card is entered as at the SPO with the exception that an additional question mark must precede the message.

NOTE

If more than one control card and/or program-parameter card is to be entered and the cards are related to the same program (e.g., an Execute card and label equation cards), the cards must all be entered as one message, using the semicolon convention to separate the cards.

Any SPO capable station may be specified to be an alternate SPO providing the station is in ready status at the time the BS message is entered. The station will remain an alternate SPO until a US message is entered for it or until the station goes not ready. A maximum of four alternate SPO's may be active at one time. If a US message has been entered for the SPO and no other SPO consoles are active, the SPO will automatically resume active status.

Input is entered from a typewriter or TWX SPO console as from the SPO. There is no input request, and the left arrow (←) is used as the END-OF-MESSAGE signal. The break key is used as an input request key when the station is typing. The station will then remain idle until input has been entered. Typing will resume with the messages queued for output. The BK message may be used to clear this queue.

Each SPO console gets only those messages it requested or which are necessary for effective system management. The following exception should be noted. If an input request is made to the SPO and the SPO is not active for output, the response to the input request will go to all SPO consoles.

ATTACHING REMOTE STATIONS TO PROGRAMS

When it is said that a remote station is attached to an object program, it means that some action has been taken to denote that input messages from that station can be read by that object program; i.e., more than one program can be attached to a given station.

If a program wishes to attach a station to itself, it may do so by performing a READ, SEEK or WRITE statement which references that station.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

If the operator at a remote station which has SPO capabilities desires to attach his station to a program, he may do so by entering an EXECUTE or RUN card for that program.

A <remote station address> is defined as:

<integer> / <integer>

where the first <integer> specifies the number of the terminal related to the remote station being referenced, and the second <integer> specifies the relevant buffer number.

A <remote station message> is defined as:

a string of characters, the end of which is recognized to be a group mark; i.e., a left arrow (←) or END OF MESSAGE.

THE RUN CARD

The RUN card is provided for use at remote data communication stations which have SPO capabilities. The purpose of the RUN card is to provide the operator of such a data communication station with the ability to attach his station to a program.

If, when a RUN card is recognized by the MCP, the designated program is in the MIX, the attaching process alone will take place and the message

<job specifier> RUNNING

will be given in response to the RUN card. However, if the specified job is not then in the MIX, the program will be scheduled and executed in the normal fashion. The station will then be attached.

The RUN card must contain the following information:

? RUN <program specifier> <comment>

Example:

? RUN MANYSTA/HANDLER

ADDITIONAL KEYBOARD INPUT MESSAGES FOR REMOTE STATIONS

The following paragraphs describe new messages made available for use at remote stations which have SPO capabilities.

THE BS MESSAGE

The BS message sets the station indicated by <remote specifier> as a SPO console.

The BS message has the following format:

BS <remote specifier>

THE HR MESSAGE

The HR message is used to detach a station from a program. It can be considered to be the opposite of a RUN card.

The HR message has the following format:

<mix index> HR

Example:

1 HR

THE LI MESSAGE

The LI message is the predecessor of a more rigorous log-in message which will be provided at a later date.

The primary purpose of the LI message is to require that a remote operator identify himself as a legitimate user of the system in order for the MCP to allow him to make use of the system.

If an LI message is entered while a remote operator is already logged in, the MCP will log-out the previous user before logging in the new user.

The current LI message has the following format:

LI

THE LO MESSAGE

The LO message is provided so that a remote user may log-off after having logged-in. This is desirable in that anyone who attempts to use the remote typewriter, subsequent to the departure of the proven legitimate user, must also be a legitimate user.

The LO message must have the following format:

LO

THE SM AND HM MESSAGES

The <mix> SM message requests MIX messages for a given job. MIX messages for other jobs attached to a station are not affected.

The SM message requests MIX messages for all jobs which originate from a control console. This request is assumed at log-in time and remains in effect until an HM message is entered.

The <mix> HM message turns off the request for MIX messages for a given job. MIX messages for other jobs attached to a station are not affected.

The HM message requests that no MIX messages be given for jobs originating from a control station. This request remains in effect until an SM message is entered or until the station is logged out and logged in again.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

The SM message may have either of the following formats:

SM

or

<mix index> SM

The HM message may have either of the following formats:

HM

or

<mix index> HM

Examples:

SM
1 SM
HM
1 HM

THE SS MESSAGE

The SS message may be used at the central SPO or, if preceded by a question mark, on a remote station with SPO capabilities to direct a message to a remote station which has SPO capabilities, or to the SPO. If the station addressed is not recognized to have SPO capabilities or is Not Ready, an INV STN message is returned. The message, as provided at the addressed station, has a prefix which includes the address of the originator.

The SS message has the following format:

SS <remote station address> : <remote station message>

or

SS SPO : <remote station message>

Examples:

SS I/O : ARE YOU THERE
?SS SPO : I NEED A SCRATCH TAPE

THE US MESSAGE

The US message un-sets the station indicated by <remote specifier> from SPO console status.

The US message has the following format:

US <remote specifier>

ALTERNATE SPO CONSOLES

Any SPO capable station may be specified to be an Alternate SPO providing that it is ready at the time the BS message is entered. It will remain an alternate SPO until a US message is entered for it or until the station goes not ready. Up to four alternate SPO's may be active at one time. There is provision made so that the SPO will automatically resume active status if there are no other active SPO's.

THE BREAKOUT PROCESS

ALGOL programs create a permanent break file on disk when a break statement is executed. COBOL programs act similarly when a RERUN statement is executed or a RERUN clause invoked with one exception: a COBOL reel RERUN clause specifies that the break file is placed on the beginning of an output reel of a tape file.

In this instance, a temporary break file is created on disk, is copied to the tape, and the disk file is destroyed on completion or termination of the copy.

The ONEBREAK option is ignored.

Tape, disk, and pseudo reader files and line print files regardless if back up label equation are permitted to be open when a break is done. Sort files, card reader, and data communication files preclude breaking. If a break is attempted while one of the forbidden files is open, the break attempt is ignored and the operator is told which file interfered with the break.

Breaks are numbered cyclically from 00 to 99. The numbering of break files following a restart is the same as would occur if no restart was done. Thus, if a break file no. 06 was used to restart a job, the next break file created by the file would be 07.

Provision has been made for the handling of write errors when the reel option is evoked, forcing the backup file to tape. If an error occurs the operator will be notified of a BADUMP, the reel is switched, and a new break file is built having the same break number.

Break files are program files named according to their break number and to whichever program built them. Thus break files created when executing the program BREAKER/A would be named BREAKER/BREAKnn where each nn is a break number. The files have thirty words per record, and the records are allocated in 500-record chunks.

Break files differ from other program files although segment zero is somewhat similar. A comment detailing segment zero's differences follows the DCMCP procedure BREAKOUT.

A copy of the job's current overlay disk, coded, and intrinsics the job might use are included in the break file. The file also has a map for exactly replacing some of the job's core areas, primarily the non-overlayable areas.

Breakout usually takes about five seconds building a two-chunk break file, during which time the system is disk bound. It uses about 1.5K of core storage.

Breaking doesn't affect the contents of files, however it is necessary to save temporary files. Both temporary and permanent disk files are entered in the directory when a break occurs. Tape files are marked to be saved.

All label equation information is retained and is used to recognize files at restart.

RESTARTING PROCESS

If the reel rerun clause was specified, the break file must be loaded from tape to disk. This is accomplished by the RS<unit> message which checks to determine if the mentioned unit is a labeled, write-enabled tape with a break file copy. If so, the system will load the copy unless there is already a permanent disk file with the same name as that of the copy. After loading, the unit is left marked and positioned for the

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

pending restart. If the load try failed, the unit is set not-in-use and locked. Note that the RS message only causes the reloading of the break files, it doesn't initiate restarts.

Files re-opened while restarting must correspond closely to those open when the break file was built, the re-opening files are therefore thoroughly checked before acceptance. A file's type may not be changed between break and reopening. File reopening is detailed below.

To restart, one simply executes or runs the appropriate break file. If the MCP is not compiled with the $ SET BREAKOUT=TRUE module the break file will be considered to be non-executable code.

Mapped areas must be replaced exactly where they were at breakout. This replacement is considered to be the "restarting" process. When replacement is complete, the operator is notified that the job has restarted, and the system begins reopening the job's files. Replacement side effects are presented below. Restarts must wait for particular areas to become available, therefore restarts should be performed with no jobs in the mix, preferably immediately after a HALT/LOAD.

All control cards used in originally initiating the program are used to restart that program stack, label equation, and common cards are ignored by the restarting process.

Once a restart job has been initiated (BOJ) and until the restarting process is complete, the system is occupied replacing or rebuilding the job's core areas. It is possible that the entire required address range is not available, e.g., that some in use area is interfering with replacement. Simply waiting resolves some interference. Some areas must be moved or denied space during restarts as it is not possible to wait so that the interference will clear up.

There are various consequences of replacement. The restart will be automatically ESed if the memory configuration differs from that at break. The IN, OT, and ST operator requests and changing intrinsics (either with a CI or XI) are not valid when applied to restarting jobs.

The operator may monitor a job's progress in restarting. If he requests <mix> WY, he will be told if the restart is waiting due to interfering areas. Rarely, however, should the MCP (mix zero) or the restarting job itself cause interference.

A restart job that does not have an excessive amount of overlay disk usually takes about four seconds restarting. None of its storage is overlayable. Of itself, a restarting job uses less core storage than it did breaking. However, when resolving interferences involves moving many areas, a NO MEM situation may occur. In this event, movement is temporarily abandoned, and the system recovers; a HALT/LOAD and new restart attempt may be required.

FILE HANDLING

After a job has restarted, its files are "reopened". The system finds files, checks them against data saved at break, and then repositions them forward accordingly. If a file checked is somehow unsuitable, the operator is told what was wrong, and the system tries finding the correct file.

Only tape files are actually spaced. The restart job is terminated if an irrecoverable parity or other problem occurs. Positioning other files is ignored; their proper positioning derives from internal references left intact.

Note that file changes made after a break aren't considered while reopening the file. For instance, errors may result if a record is added, deleted or altered. Such errors need not appear immediately after the breakpoint.

A job's MT, DK, CD, LP, PBT, and PBD files may be open at break. Reopening these type files is considered below.

Note carefully the checks which apply. To break at all, other type files and sort files must have been closed. They are ignored while the system reopens files.

LP, PBT, and PBD files need no attention while breaking. At restart, lineprint continues with whichever line logically followed the break. The line printer selected isn't positioned within the page or physically labeled. Backup files restart similarly, but with new files. Only type correspondence is checked.

Disk files are permanent-or-not according to whether they are mentioned in the directory at break; temporary files become permanent. Their save factor is then set to sixty-three if it was previously zero or unspecified. The directory's information about permanent files in use may be inaccurate. Therefore, it is updated each break.

At restart, disk files are checked for blocking, record size, number and use of rows, allowed number of rows, and end-of-file suitability.

Pseudoreaders are checked for type correspondence only. The system's normal checks detect short control-decks, and the other disk file checks aren't relevant. However, control decks have internal control cards linkages, references to which reopening does not check. Further, the system can not find a sub-deck unless it is current in a pseudoreader.

Tape files (NOT PBT) have a physical block count recorded each break. Finding an input tape file may involve searching a multifile. Reopening tape files are found as though they are input files; once checked and accepted they are spaced forward according to the difference between current and break counts.

Thus, a tape is effectively positioned relative to its reels' beginning.

As accepted, tapes are positioned in parallel.

If an output tape reel was RS-ed to load the break just restarted, then the tape was checked by the RS handler, and its unit was left not-in-use, positioned and marked to prevent jobs which are not restarts from finding it. The unit is therefore checked for marks the RS handler left; if the unit is unmarked, a tape is found, checked like other tapes, and also checked for a break file copy.

Other tapes (not PBT) are checked for type, label, "format", dump, and write-enable correspondence; Tape "format" correspondence is wrong if the file requested is found after the block broken. Tape label correspondence is wrong if the broken file had (lacked) a label and the file found lacked (had) one. Dump correspondence is wrong if the file found locks a needed break file copy. Write-enable correspondence is wrong if the file needs (lacks) a write-ring.

SYSTEMS EFFECTS

Breakout-restart was redesigned to eliminate its characteristic maintenance requirement. With this end in view, the new design confines its attention to the job principally involved, thereby markedly reducing tangential interactions with the system and other jobs. Two of such interactions left warrant attention.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

Note that breakout saves all intrinsics the job may use. This ensures the integrity of references thereto, but users may occasionally restart with functionally dated intrinsics, an error. To avoid potential problems, the same MCP and intrinsics used at breakout time should be used at restart time.

Also, note that restart storage replacement involves moving areas. Such a movement must only occur when all extant references to the area can be corrected. Therefore, any MCP changes must be made cooperative with Breakout/Restart requirements. If one loses control with a new reference to such an area pending or if one introduces new areas, one should consider whether and how they should move.

BREAKOUT MESSAGES

1) "--CAN-T BREAK <data file designator> <rdc> : <job specifier>".
The job tried breaking with a file of unsuitable type open. The break try is ignored.

2) "<priority> : <job specifier>=<mix> : BREAK<break number> BUILT".
The specified job just broke, creating the break file <program name> / BREAK <break number>.
The break file is then moved to an output tape, if necessary.

3) "<priority> : <job specifier>=<mix> : BADUMP ON <unit>".
The system couldn't copy a break file onto the tape mentioned. It will try again on a new reel.

THE RS MESSAGE

This allows the operator to add a break file to the directory, the break file having been copied to a Cobol job's output tape. The RS message format is:

"RS<unit>".

The responses are:
1) "RS<unit>INV KBD". The unit isn't a tape.
2) "<unit> <note>". The note is one of NOT READY, IN USE, SCRATCH, WRITE LOCK, or NO DUMP.
The unit must be an available, labeled, write-enabled tape with a break file copy.
3) ".<program name>/BREAK<break number> NOT ADDED.DUP LIB RS <unit>."
There is already a disk file with the names mentioned. The break file on the unit isn't loaded.
4) "<unit> ERROR IN DUMP". The tape-disk break file copy went awry.
The break file isn't loaded.
5) "<program name>/BREAK <number> ADDED. TAPE POSITIONED: <unit>."
The RS<unit> was successful. The unit is left positioned and marked for the pending restart of the loaded break file.

RESTART MESSAGES

1) "<job specifier>=<mix index> GONE <time>."
The job was a restart which was ES-ed or DS-ed before having restarted.

2) "<priority>:<job specifier>=<mix index> RESTARTED."
The designated restart job was completed replacing core storage and now has a normal job structure, i.e., it has "restarted". The job will begin reopening files next.

3) "<priority>:<job specifier>=<mix> RESTART IS <state>".
State is WAITING or MOVING.
The operator requested <mix> WY before the job restarted. This response tells what restart is doing.

4) "--MIX: <mix>,...,<mix> IN THE WAY". The mixes (e.g., zero for the MCP) are using core areas needed to replace the restarting job's storage.

FILE REOPENING MESSAGES

1) "--WRONG FILE <data file designator> <rdc>:<job specifier>".
Reopening involves checking files for compatibility with those in use at breakout. The designated file is not sufficiently compatible; the next message hints why. The operator may respond with an OK, WY, or DS reply, OK initiating a recheck.

2) "--WRONG <hint>,...,<hint>."
The hints are among the following:
A. WRITE STATE -- The file's write ring is in the wrong place (in the box or on the tape).
B. LABEL -- The file lacks (needs) a label.
C. TYPE -- A file on disk (tape, line-print) at break must be on disk (tape, line-print) at restart. For example, a LP file was broken and the operator tried reopening it as a PBD.
D. ROWS USED -- Disk files have up to twenty rows. The one found doesn't have the right one.
E. NO. OF ROWS - The disk file found has the wrong number of rows allowed.
F. EOF -- It's too short.
G. ROW LENGTH -- Its rows are the wrong size.
H. FORMAT -- If it's a disk file, it's blocking or record length is wrong. If it's a tape file, it was found beyond where it was in Breakout.
I. SECURITY -- A security error occurred.
J. NO DUMP -- The tape file lacks a break file copy.

EXTENDED ALGOL SYNTACTICAL ERROR MESSAGES

Error No.	Routine	Error Message
000	Block	Declaration not followed by semicolon.
001	Block	Identifier declared twice in same block.
002	PROCEDURE Declaration	Specification PART contains identifier not in format parameter PART.
003	Block	Nonidentifier in identifier LIST of declaration.
004	PROCEDURE Declaration	STREAM PROCEDURE declaration preceded by illegal declarator.
005	PROCEDURE Declaration	PROCEDURE declaration preceded by illegal declarator.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

006	PROCEDURE Declaration	PROCEDURE identifier repeated in same block (not FORWARD).
007	PROCEDURE Declaration	PROCEDURE identifier not followed by left parenthesis or semicolon in PROCEDURE declaration.
008	PROCEDURE Declaration	Formal parameter LIST not followed by right parenthesis.
009	PROCEDURE Declaration	Formal parameter part not followed by semicolon.
010	PROCEDURE Declaration	VALUE PART contains identifier not in formal parameter LIST.
011	PROCEDURE Declaration	VALUE PART not ended by semicolon.
012	PROCEDURE Declaration	Missing or illegal specification PART.
013	PROCEDURE Declaration	OWN used in ARRAY specification.
014	PROCEDURE Declaration	SAVE used in ARRAY specification.
016	ARRAY Declaration	ARRAY identifier not followed by left bracket.
017	ARRAY Declaration	Lower bound in ARRAY declaration not followed by colon.
018	ARRAY Declaration	Bound pair in ARRAY declaration not followed by right bracket.
019	ARRAY Specification	Illegal lower bound designator in ARRAY specification.
020	Block	OWN immediately before identifier (no type) in declaration.
021	Block	SAVE immediately before identifier (no type) in declaration.
022	Block	STREAM immediately before identifier (the word PROCEDURE was left out).
023	Block	Declarator illegally preceded by another declarator.
024	PROCEDURE Declaration	LABEL passed to a function.
025	Block	Declarator or specifier illegally preceded by OWN, SAVE or another declarator.
026	FILE Declaration	Missing left parenthesis in FILE declaration.
027	FILE Declaration	MISSING RECORD SIZE.

028	FILE Declaration	Illegal buffer part or SAVE factor in FILE declaration.
029	FILE Declaration	Missing right parenthesis in FILE declaration.
031	LIST Declaration	Missing left parenthesis in LIST declaration.
032	FORMAT Declaration	Missing left parenthesis in FORMAT declaration.
033	SWITCH Declaration	SWITCH declaration does not have ← or FORWARD after identifier.
034	SWITCH FILE Declaration	Missing ← after SWITCH FILE identifier.
035	SWITCH FILE Declaration	NON-FILE identifier in declaration of SWITCH FILE LIST.
036	SWITCH FORMAT Declaration	SWITCH FORMAT identifier not followed by ←.
037	SWITCH FORMAT Declaration	Missing left parenthesis at start of SWITCH FORMAT LIST.
038	SWITCH FORMAT	SWITCH FORMAT segment > 1022 words.
039	Block	Number of nested blocks > 31.
040	I/O Declaration	Program parameter block size exceeded.
041	SWITCH LIST	Missing after SWITCH LIST ID.
042	SWITCH LIST	Illegal list ID appearing in SWITCH LIST.
043	I/O Declaration	Missing right bracket after DISK in FILE Declaration.
044	I/O Declaration	Missing left bracket after DISK in FILE Declaration.
045	"DEFINE DEC"	Missing "=" after defined identifier.
046	Variable	Non-literal array bound not global to array declaration.
047	Variable	Item following @ not an integer.
048	PROCEDURE Declaration	The number of parameters does not agree with the number of parameters in the FORWARD declaration.
049	PROCEDURE Declaration	The type of this parameter does not agree with its type as given in the FORWARD declaration.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

050	PROCEDURE Declaration	The value part differs from the value part of the FORWARD declaration. The formal parameter of the FORWARD declaration and the corresponding parameter in the actual declaration are specified respectively as call-by-name and call-by-value, or vice versa.
059	ARRAY Declaration	Improper ARRAY size.
060	FAULT Statement	Missing ← in FAULT Statement.
061	FAULT Declaration	Invalid FAULT Type: Must be FLAG, EXPOVR, ZERO, INTOVR, or INDEX.
070	CASE Statement	Missing BEGIN.
071	CASE Statement	Missing END.
100	Anywhere	Undeclared identifier.
101	PROCEDURES	An attempt has been made to address an identifier which is local to one procedure and global to another. If the quantity is a procedure name or an OWN variable, this restriction is relaxed.
102	Arithmetic Expression	Conditional expression not of arithmetic type.
103	Primary	Primary may not begin with this type quantity.
104	Anywhere	Missing right parenthesis.
105	Anywhere	Missing left parenthesis.
106	Primary	Primary may not start with declarators.
107	Boolean Expression	The expression is not of Boolean type.
108	Expression	Relation may not have conditional expressions as arithmetic expressions.
109	Boolean Expression	Primary is not of Boolean type.
110	Boolean Expression	Non-Boolean operator in Boolean expression.
111	Boolean Expression	No expression (arithmetic, Boolean, or designational) may begin with this type quantity.
112	Boolean Expression	No expression (arithmetic, Boolean, or designational) may begin with a declarator.

113	Anywhere	Either syntax or range of literals for concatenate operator is incorrect.
114	Partial Word	Either syntax or range of literals for partial word designator is incorrect.
115	Designational Expression	Expression not of designational type.
116	IF Clause	Missing THEN.
117	Anywhere	Missing left bracket.
118	Anywhere	Missing right bracket.
119	Compound Tail	Missing semicolon or END.
120	Compound Tail	Missing END.
121	Actual Parameter Part	An indexed FILE may be passed by name only - and only to STREAM PROCEDURE. STREAM PROCEDURE may not RELEASE this type parameter.
122	Actual Parameter Part	Expressions may not pass by name to STREAM PROCEDURES.
123	Actual Parameter Part	Actual and formal parameters not same type.
124	Actual Parameter Part	Actual and formal arrays not same number of dimensions.
125	Actual Parameter Part	STREAM PROCEDURE may not be passed as an actual parameter to PROCEDURE.
126	Actual Parameter Part	Actual parameter may not begin with this type quantity.
127	Actual Parameter Part	This type quantity may not be passed to STREAM PROCEDURE.
128	Actual Parameter Part	Actual and formal parameters do not agree in number or extra right parenthesis.
129	Actual Parameter Part	Illegal parameter delimiter.
130	RELEASE Statement	No FILE name.
131	DO Statement	Missing UNTIL.
132	WHILE Statement	Missing DO.
133	LABEL	Missing colon in LABEL.
134	LABEL	LABEL not declared in this block.
135	LABEL	LABEL has already occurred.

PRINTED IN U.S. AMERICA REVISED 8/70

1031966

136	FORMAT	Improper FORMAT editing phrase.
137	FORMAT	FORMAT editing phrase does not have integer where required.
138	FORMAT	Width too small in E or F editing phrase.
139	Table	DEFINE nested more than 8 deep.
140	FORMAT	Integer in FORMAT > 1023.
141	Scanner	Integer or identifier more than 63 characters.
142	DEFINE	DEFINE more than 2047 characters (blank suppressed).
143	Compound Tail	Extra END.
144	Statement	Statement may not start with this type identifier.
145	Statement	Statement may not start with this type quantity.
146	Statement	Statement may not start with a declarator. (It may be a missing END of a PROCEDURE or a misplaced declaration.)
147	SWITCH	More than 256 expressions in SWITCH declaration.
148	PRT Space	More than 1023 program reference table cells required for this program.
149	PRT Space	More than 255 stack cells required for this PROCEDURE.
150	Actual Parameter Part	Constants may not be passed by name to STREAM PROCEDURES.
152	FOR Statement	Missing ← following INDEX variable.
153	FOR Statement	Missing UNTIL or WHILE in STEP Element.
154	FOR Statement	Missing DO in FOR clause.
155	IF Statement	Missing ELSE.
156	LIST Element	Designational expression may not be LIST element.
157	LIST Element	Row designator may not be LIST element.
158	LIST Element	Missing right bracket in elements.
159	PROCEDURE Statement	Illegal use of PROCEDURE or function identifier.

160	Purge	Declared LABEL did not occur.
161	Purge	Declared FORWARD PROCEDURE did not occur.
162	Purge	Declared FORWARD SWITCH did not occur.
163	FORMAT	Width of field more than 63 characters.
164	ZIP Statement	Missing comma in ZIP or WAIT Statement.
200	Emit	Segment too large (> 4093 syllables).
201	Simple Variable	Partial word designator not leftmost in left part LIST.
202	Simple Variable	Missing . or ←.
203	Subscripted Variable	Wrong number of subscripts in row designator.
204	Subscripted Variable	Missing right bracket in row designator.
205	Subscripted Variable	Row designator outside of actual parameter in LIST or FILL statement.
206	Subscripted Variable	Missing right bracket.
207	Subscripted Variable	Missing left bracket.
208	Subscripted Variable	Wrong number of subscripts.
209	Subscripted Variable	Partial word designator not leftmost in left part LIST.
210	Subscripted Variable	Missing . or ←.
211	Variable	PROCEDURE identifier appears outside of scope in left part.
212	Variable	Sub-array designator permitted as actual parameter only.
250	STREAM Statement	Illegal STREAM statement.
251	STREAM Statement	Missing ←.
252	Index	Missing + or -.
253	Index	Missing number or STREAM variable.
255	Destination String	Missing string in DS ← LT statement.
256	RELEASE Statement	Missing parenthesis; or, FILE identifier not a formal parameter.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

257	GO TO, LABEL or JUMP Statement	LABEL specified not the same nest level as preceding appearance of the LABEL.
258	LABEL	Missing colon.
259	LABEL	LABEL appears more than once.
260	GO TO Statement	Missing LABEL in GO TO or JUMP OUT TO statement.
261	JUMP Statement	Missing OUT in JUMP OUT statement.
262	Nests	Missing parenthesis.
263	IF Statement	Missing SC in IF Statement.
264	IF Statement	Missing relational in IF Statement.
265	IF Statement	Missing ALPHA, DC or string in IF Statement.
266	IF Statement	Missing THEN in IF statement.
267	GO TO Statement	LABEL undefined in GO TO statement.
268	Emit Literal	Repeat index $\geq$ 64 was specified; or, too many formal parameters, locals and labels.
269	Table	Constant specified too large or too small.
270	IF Statement	Relational operator other than = in test <source for ALPHA>.
271	IF Statement	Improper construct for <source with literal>.
281	DOUBLE Statement	Missing left parenthesis.
282	DOUBLE Statement	Too many operators.
283	DOUBLE Statement	Too many operands.
284	DOUBLE Statement	Missing comma.
285	DOUBLE Statement	Missing right parenthesis.
286	DOUBLE Statement	Illegal parameter.
300	FILL Statement	Identifier following FILL not ARRAY identifier.
301	FILL Statement	Missing WITH in FILL statement.
302	FILL Statement	Improper FILL element.
303	FILL Statement	Nonoctal character in octal FILL. The three low-order bits are converted and compilation continues.

304	FILL Statement	Improper row designator.
350	Check comma	Missing or illegal parameter delimiter in SORT or MERGE statement.
351	Output	Illegal TYPE for SORT or MERGE output procedure.
352	Output	Output procedure in SORT or MERGE statement does not have exactly two parameters.
353	Output	First parameter of output procedure must be BOOLEAN.
354	Output	Second parameter of output procedure must be ONE-DIMENSION ARRAY.
355	SORT Statement	Missing left parenthesis.
356	HIGHVALUE	Illegal TYPE for SORT or MERGE HIGHVALUE procedure.
357	HIGHVALUE	HIGHVALUE procedure does not have exactly one parameter.
358	HIGHVALUE	HIGHVALUE procedure parameter is not ONE-DIMENSION ARRAY.
359	COMPARE	SORT or MERGE COMPARE procedure NOT BOOLEAN.
360	COMPARE	COMPARE procedure does not have exactly two parameters.
361	COMPARE	COMPARE procedure first parameter not ONE-DIMENSION ARRAY.
362	COMPARE	COMPARE procedure second parameter not ONE-DIMENSION ARRAY.
363	PROCEDURE	SORT statement input procedure not BOOLEAN.
364	PROCEDURE	Input procedure does not have exactly one parameter.
365	PROCEDURE	Input procedure parameter is not a ONE-DIMENSION ARRAY.
366	SORT Statement	Missing right parenthesis.
367	MERGE Statement	Missing left parenthesis.
368	MERGE Statement	More than 7 or less than 2 files to merge.
369	MERGE Statement	Missing right parenthesis.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

400	MONITOR Declaration	Missing FILE identifier in MONITOR declaration.
401	MONITOR Declaration	Missing left parenthesis in MONITOR declaration.
402	MONITOR Declaration	Improper subscript for MONITOR LIST element.
403	MONITOR Declaration	Improper subscript expression delimiter in MONITOR LIST element.
404	MONITOR Declaration	Improper number of subscripts in MONITOR LIST element.
405	MONITOR Declaration	LABEL or SWITCH monitored at improper level.
406	MONITOR Declaration	Improper MONITOR LIST element.
407	MONITOR Declaration	Missing right parenthesis in MONITOR declaration.
408	MONITOR Declaration	Improper MONITOR declaration delimiter.
409	DUMP Declaration	Missing FILE identifier in DUMP declaration.
410	DUMP Declaration	Missing left parenthesis in DUMP declaration.
411	DUMP Declaration	Subscripted variable in DUMP LIST has wrong number of subscripts.
412	DUMP Declaration	Subscripted variable in DUMP LIST has wrong number of subscripts.
413	DUMP Declaration	Improper ARRAY DUMP LIST element.
414	DUMP Declaration	Illegal DUMP LIST element.
415	DUMP Declaration	More than 100 labels as DUMP LIST elements in 1 DUMP declaration.
416	DUMP Declaration	Illegal DUMP LIST element delimiter.
417	DUMP Declaration	Missing DUMP LABEL in DUMP declaration.
418	DUMP Declaration	Missing colon in DUMP declaration.
419	DUMP Declaration	Improper DUMP declaration delimiter.
420	READ Statement	Missing left parenthesis in READ statement.
421	READ Statement	Missing left parenthesis in READ REVERSE statement.

422	READ Statement	Missing FILE in READ statement.
424	READ Statement	Improper FILE delimiter in READ statement.
425	READ Statement	Improper FORMAT delimiter in READ statement.
426	READ Statement	Improper delimiter for second parameter in READ statement.
427	READ Statement	Improper row designator in READ statement.
428	READ Statement	Improper row designator delimiter in READ statement.
429	READ Statement	Missing row designator in READ statement.
430	READ Statement	Improper delimiter preceding LIST in READ statement.
433	Action Label	Missing right bracket in READ or SPACE statement.
434	SPACE Statement	Missing left parenthesis in SPACE statement.
435	SPACE Statement	Improper FILE identifier in SPACE statement.
436	SPACE Statement	Missing comma in SPACE statement.
437	SPACE Statement	Missing right parenthesis in SPACE statement.
438	WRITE Statement	Missing left parenthesis in WRITE statement.
439	WRITE Statement	Improper FILE identifier in WRITE statement.
440	WRITE Statement	Improper delimiter for first parameter in WRITE statement.
441	WRITE Statement	Missing right bracket in <carriage control part> of WRITE statement.
442	WRITE Statement	Illegal carriage control delimiter in WRITE statement.
443	WRITE Statement	Improper second parameter delimiter in WRITE statement.
444	WRITE Statement	Improper row designator in WRITE statement.
445	WRITE Statement	Missing right parenthesis after row designator in WRITE statement.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

446	WRITE Statement	Missing row designator in WRITE statement.
447	WRITE Statement	Improper delimiter preceding LIST in WRITE statement.
448	WRITE Statement	Improper LIST delimiter in WRITE statement.
449	READ Statement	Improper LIST delimiter in READ statement.
450	LOCK Statement	Missing left parenthesis in LOCK statement.
451	LOCK Statement	Improper FILE in LOCK statement.
452	LOCK Statement	Missing comma in LOCK statement.
453	LOCK Statement	Improper <unit disposition part> in LOCK statement.
454	LOCK Statement	Missing right parenthesis in LOCK statement.
455	CLOSE Statement	Missing left parenthesis in CLOSE statement.
456	CLOSE Statement	Improper FILE in CLOSE statement.
457	CLOSE Statement	Missing comma in CLOSE statement.
458	CLOSE Statement	Improper <unit disposition part> in CLOSE statement.
459	CLOSE Statement	Missing right parenthesis in CLOSE statement.
460	REWIND Statement	Missing left parenthesis in REWIND.
461	REWIND Statement	Improper <FILE part> in REWIND statement.
462	REWIND Statement	Missing right parenthesis in REWIND.
463	Block	MONITOR declaration in specification of PROCEDURE.
464	Block	DUMP declaration in specification of PROCEDURE.
465	DUMP	DUMP indicator must be unsigned integer or simple variable.
500	SEARCHLIB	Illegal LIBRARY identifier.
501	SEARCHLIB	LIBRARY identifier not in directory.
502	SEARCHLIB	Illegal LIBRARY start point.

503	SEARCHLIB	Separator required between start point and length.
504	SEARCHLIB	Illegal LIBRARY length.
505	SEARCHLIB	Missing bracket.
507*	SEARCHLIB	Magnetic tape positioning error.

* Although this is actually the result of a hardware malfunction, it is detected by the compiler and is therefore emitted as a Syntax Error Message. The program will not compile properly from this point on, but compilation will continue. Try putting the Subprogram Library Tape on a different unit.

COBOL COMPILER ERROR AND DIAGNOSTIC MESSAGES

ACCESS MISSING

ACT. KEY QUALIFICATION ILLEGAL

ACTUAL KEY must be a 77 level CMP or CMP-1 item.

ACT. KEY MISSING

ACT. KEY SIZE ILLEGAL

ACTUAL KEY > 11 digits.

ACT. KEY TYPE ILLEGAL

ACTUAL KEY must be elementary.

ACT. KEY USAGE ILLEGAL

ACTUAL KEY must be a COMPUTATIONAL or COMPUTATIONAL-1 elementary item in WORKING-STORAGE.

ADD NO ELEMENT ITEMS

ARITHMETIC OPERAND CLASS XXXXX

Data-name "XXXXX" should be an arithmetic operand, but its CLASS IS INCORRECT.

ARRAY SIZE ERROR STATEMENT TRUNCATION

Too many list elements in a diagnostic statement.

PRINTED IN U.S. AMERICA REVISED 8/70

1031966

ASSIGN SYNTAX ERROR

BUFFER MISSING

A disk file has been reserved declaring no alternate areas.

BY MISSING

The word BY is missing in the PERFORM statement.

XXX CHARACTERS MISSING

A COMPUTATIONAL item in the record description is not word oriented or a record in a file is not a multiple of eight characters in length. The Compiler will insert FILLER to make the item start at the beginning of a word, thus changing record total size.

NN CHARACTERS MISSING

Description does not match word boundary.

CARD TRUNCATION XXXXX

CARDS nnnnn

CHECK RECORD SIZE

This message is due to:

A. RECORD SIZE declared differs from SIZE in record description.
B. Character SIZE in BLOCK CONTAINS is not integer multiple of record size.

CLASS DECLARATION ERROR

Misplaced 77 level item.

CLASS ERROR

In MEMORY SIZE clause, size is not given as an integer. (Should be given as a number of words rather than a number of memory modules.)

CLASS ERROR

The CLASS of the item is not declared with an acceptable reserved word: NUMERIC, ALPHABETIC, ALPHANUMERIC, or AN.

CLASS ERROR XXXXX

The CLASS of the data (XXXXX) is either:

A. Not numeric or an edited numeric (arithmetic statement), or
B. It is an invalid receiving field for a MOVE statement.

CLASS ERROR ILLEGAL OPERAND

The operand in this statement has a CLASS which is not legal in the context.

COMPILE ERROR

If occurs after all other diagnostics have been removed, send SAR and source deck.

COMPILE ERROR 01347000

A data-name has been defined more than 125 times.

COMPILE O.K. B 5500 MO-DA-YR

A terminating message signifying that a successful compilation has been completed, as opposed to a did not compile. Certain warning messages, given by the Compiler, may be shown without affecting the compilation. This message does not imply that the program is logically correct.

COMPILE TIME NNNNN SEC.

Information on compile time.

CONDITIONAL GROUP SIGNED XXXXX

The comparison operand contains a signed item in the group identified by XXXXX.

CONDITIONAL GROUP SIZE XXXXX

The group identified by XXXXX contains an item of variable size or the groups are of different size.

CONDITIONAL GROUP USAGE XXXXX

The group identified by XXXXX contains an item of COMPUTATIONAL usage.

CONDITIONAL LITERAL OPERAND SIZE

The limit of 63 characters length has been exceeded.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

CONDITIONAL NAME ERROR

Condition-name must not be a reserved word.

CONDITIONAL OPERAND CLASS ERROR

A numeric item is not allowed in this statement, or this is a comparison of signed and unsigned items.

CONDITIONAL OPERAND SIGNED ERROR

Comparison operand is a signed numeric.

CONDITIONAL OPERAND USAGE ERROR

Comparison of group items containing CMP items.

CONDITIONAL SPECIFICATION SIZE ERROR

CONDITIONAL VALUE SIZE ERROR

88 level on an item of more than 63 characters.

COPY LEVEL ERROR

The level-number is beyond range 0 thru 49 due to incrementation during a copy.

(CORRESPONDING) XXXXX (DATA NAME) OF XXXXX (FILE NAME)

This message itemizes all corresponding items being acted upon by a corresponding operation.

DECLARATION ERROR

The RENAMING option is used, but the file-name is not shown in a prior SELECT statement.

DID NOT COMPILE B 5500 MO-DA-YR

DISK SIZE NNNNN

Requested Information.

DUPL. FILE NAME

Duplicated file-name in FD, MD, or SD entry.

DUPL. $$ CARD

DUPLICATE LABEL

The name is a duplicate of the one previously defined. OBJECT-COMPUTER specified two or more times. (Warning only.)

ERRORS nnnnn

EXTRA ARITHMETIC OPERAND XXXXX

Several data-names are shown following the TO. One of the data-names requires qualification.

EXTRA FILE DECLARATION ERROR

More than 50 files have been declared for a file and/or the number of files times 18, plus the length of all the file-names in characters, plus 32 is greater than 1024. Note that assign to sort disk is two files and assign to sort disk and three sort-tapes is five files.

FILE DECLARATION ERROR XXXXX

The MONITOR or DUMP statement does not declare a file for the printer.

FILE NOT SELECTED

This message is due to:

A. Compiler is expecting the word SELECT as the next word.
B. The file-name is not shown in a SELECT statement in the ENVIRONMENT DIVISION.
C. No files have been SELECTed.

FROM MISSING PERFORM STATEMENT

The word FROM is missing in the PERFORM statement.

FROM MISSING XXXXX

GROUP CONDITIONAL OPERAND SIZE

The size of an elementary item and a group item in a comparison is different, or one of the items in the comparison is variable-length.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

GROUP CONDITIONAL OPERAND USAGE

A COMPUTATIONAL ITEM is contained in the comparison between a group and an elementary item.

GROUP PICTURE SPECIFICATION SYNTAX ERROR

PICTURE cannot be used at the group level.

GROUP SIZE ERROR

The sum of the SIZE of each elementary item does not agree with the SIZE stated at the group item level. The Compiler continues, using the sum of elementary item sizes, or a record is not a multiple of eight characters.

H-DATA-IMPROPER FOR TSS

HIERARCHY GROUP LEVEL ERROR

The level-number is illegal, it does not match a previously defined group level-number.

HYPHEN SPELLING ERROR

ILLEGAL ARITHMETIC CLASS XXXXX

The data-name shown does not have the proper CLASS for use as an arithmetic operand.

ILLEGAL ARITHMETIC LITERAL XXXXX

The literal shown is a non-numeric literal and cannot be used on an operand in an arithmetic statement.

ILLEGAL ARITHMETIC OPERAND XXXXX

This message is due to:

A. A literal may only be preceded by a + or -, or
B. The symbol should be plus or minus, or
C. A word is spelled incorrectly, or used illegally.

ILLEGAL ASSIGNMENT FD XXXX

ILLEGAL ASSIGNMENT MD XXXX

ILLEGAL ASSIGNMENT SD XXXX

The file assignments in the FILE-CONTROL section of the source program have been checked against the file descriptions in the DATA DIVISION and a conflict has been found. For example, a file assigned to DISK has been described with an FD in the FILE section rather than an MD.

ILLEGAL ASSIGNMENT SPECIFICATION

ILLEGAL BLOCK SIZE SPECIFICATION

The BLOCK SIZE for magnetic tape files is specified to be greater than 1023 words.

ILLEGAL CLASS DECLARATION

A sign has been specified for a non-numeric field or editing has been requested on a non-numeric item.

ILLEGAL CLASS SIZE DEPENDING OPERAND

The DEPENDING ON operand is not an unsigned integer; a numeric field is required.

ILLEGAL CLASS SIZE DEPENDING OPERAND FILE XXXXX

Numeric item is required.

ILLEGAL CLASS SIZE DEPENDING OPERAND RECORD XXXXX

Numeric item is required. No character count is specified for TECHNIQUE-B or TECHNIQUE-C records.

ILLEGAL CLASS SPECIFICATION

ILLEGAL COMPILE OPERATOR

Debugging compile MNEMONIC operator cannot be found.

ILLEGAL CONDITIONAL OPERAND

The message is the result of an illegal Amount Comparison operand, or a literal in a condition having the wrong CLASS.

ILLEGAL CONDITIONAL OPERAND XXXXX

This is an error in the relation shown in XXXXX, or the relation itself is incomplete, or XXXXX may be undefined.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

ILLEGAL CONDITIONAL OPERATOR XXXXX

There is a missing Relational Operator.

ILLEGAL COPY

This is a COPY of a group item which includes this COPY entry.

ILLEGAL DECLARATION

A numeric item JUSTIFIED LEFT must be an integer, no scaling is allowed.

ILLEGAL DUPL. FILE NAME

File-names must be unique.

ILLEGAL DUPL. NAME XXXXX

The data-name given is a duplicate. XXXXX was previously used as a synonym.

ILLEGAL DUPL. SPECIFICATION

An item is described within a POINT LOCATION clause and a PICTURE.

ILLEGAL FILE INPUT-OUTPUT USAGE SPECIFICATION

There is more than one record per block in a file declared as unblocked.

ILLEGAL FILE NAME

File-name missing in a SEEK.

ILLEGAL FILE SIZE SPECIFICATION

The number of characters is greater than 1023 words.

ILLEGAL FILE TYPE XXXXX

A diagnostic statement refers to a file with other than TECHNIQUE-A or unblocked records.

ILLEGAL FROM RECORD XXXXX

A WRITE FROM can only be used on an 01 level record.

ILLEGAL GO TO DEPENDING OPERAND

The DEPENDING operand must be an integer.

ILLEGAL GROUP NAME XXXXX

This error occurs because:

A. It is not legal to move an elementary numeric or edited numeric item into a group field.
B. XXXXX should not be a group item for the MOVE in process.
C. A group item appears in a formula.

ILLEGAL GROUP OCCURS XXXXX

There is a group item with OCCURS in a diagnostic statement.

ILLEGAL INPUT-OUTPUT INTEGER

The reel number is greater than three digits.

ILLEGAL INPUT-OUTPUT SPECIFICATION

A. INVALID KEY missing from WRITE to a file assigned to disk.
B. Output file declared OPTIONAL.
C. TECHNIQUE missing when BLOCK CONTAINS > 1.
D. A file-name or diagnostics are missing.
E. OPEN OUTPUT 1 or CLOSE 1 on file assigned to disk.

ILLEGAL INTEGER OPERAND

ILLEGAL INTEGER SORT SPECIFICATION

The clause RESERVE n ALTERNATE AREAS is used with a sort-file (SD entry), and n is other than 1.

ILLEGAL INTO RECORD XXXXX

The object of a READ INTO clause must be an 01 record-name, XXXXX is not for an 01 record-name.

ILLEGAL LABEL XXXXX

The only paragraphs that may be ALTERed are those which contain only a GO TO statement. Subject paragraph is not in that category.

ILLEGAL LABEL OPERAND

The word beginning in Column 8 of the card is not allowed there.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

ILLEGAL LABEL USAGE XXXXX

The label given is either a reserved word or a data-name, or is a non-unique or illegal reference to DECLARATIVES.

ILLEGAL LITERAL XXXXX

The receiving field in a MOVE statement cannot be a literal.

ILLEGAL LITERAL CONDITIONAL XXXXX

The literal following the ALL is a non-integer numeric literal.

ILLEGAL MOVE OPERAND CLASS

The operand of the MOVE attempts to place the wrong CLASS of data in receiving field.

ILLEGAL MOVE OPERAND XXXXX

An improper MOVE was made (e.g., an ALPHABETIC, ALPHANUMERIC, or an edited numeric field into a numeric field).

ILLEGAL MOVE RECORD XXXXX

ILLEGAL MOVE USAGE OPERAND

ILLEGAL NAME XXXXX

This is usually a reserved word used incorrectly, or data name > 30 characters, or an integer used as a data name.

ILLEGAL OCCURS LEVEL

An OCCURS clause is used illegally at the 01 level.

ILLEGAL OCCURS USAGE XXXXX

An item in the Diagnostic List is an elementary OCCURS item.

ILLEGAL OCCURS VALUE DECLARATION

A VALUE caluse is given for a field requiring subscripting, or the variable size DEPENDING ON option is used with a VALUE clause.

ILLEGAL OPERAND NAME XXX

Not proper item for arithmetic.

ILLEGAL OPERAND XXXXX:

The data-name XXXXX:

A. Is not allowed in the arithmetic statement, or
B. Is other than an elementary NUMERIC data item being varied in the PERFORM statement, or
C. Should be a Figurative Constand other than END.
D. Should be an elementary item with DISPLAY USAGE.
E. When XXXXX is), subscripting may be missing or incomplete.

ILLEGAL OPERAND WRITE STATEMENT

The integer associated with CHANNEL or LINES (e.g., NN LINES) is not an unsigned integer or is not a NUMERIC data-name with unsigned integer value.

ILLEGAL OPERATOR XXXXX

The XXXXX represents the data-name in the specification OUTPUT REVERSE.

ILLEGAL PICTURE SIZE DECLARATION

The PICTURE specifies the repetition of more than 127 occurrences of the symbol.

ILLEGAL PICTURE SIZE SPECIFICATION

The PICTURE is greater than 120 characters.

ILLEGAL PROCEDURE DIVISION MISSING END DEC

The end declarative terminator is not present.

ILLEGAL PROCEDURE SPECIFICATION

ILLEGAL PROGRAM IDENT

The actual program-id does not have quotes surrounding the entry.

ILLEGAL QUALIFICATION

Incomplete, incorrect, or missing qualification. Word used as qualifier may be a reserved word.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

ILLEGAL QUALIFICATION XXXXX

When in a Synonym Construct, the Synonym must be unique such that qualification is never required nor allowed. The Synonym shown is not unique.

ILLEGAL RECORD SIZE

The size of the record exceeds 1023 words (8184 characters). 77 level > 1023 words.

ILLEGAL RECORD SIZE XXXXX

The Diagnostic Statement record SIZE must be at least 15 words (120 characters).

ILLEGAL RECORD SIZE DECLARATION

SIZE of a record > 1023 words.

ILLEGAL RECORD SIZE SPECIFICATION

The SIZE of a record for magnetic tape exceeds the limit of 1023 words (8184 characters); the size is not 80 characters for punch.

ILLEGAL RECORD SPECIFICATION

ILLEGAL RENAMES OPERAND

The data-name given in the RENAMES statement does not appear in the preceding record description.

ILLEGAL SIZE DECLARATION

A numeric item is defined to have more than 63 integer places.

ILLEGAL SIZE OPERAND XXXXX

In the PERFORM statement, the data-name represented by XXXXX is not allowed to have more than 11 characters.

ILLEGAL SIZE SPECIFICATION XXXXX

XXXXX is a double-precision field (more than 11 digits) and is illegal in:

A. A COMPUTE statement.
B. The argument furnished to an intrinsic function.
C. A formula.

ILLEGAL SIZE/USAGE SPECIFICATION

A COMPUTATIONAL item is greater than 18 digits in length. Either the SIZE or USAGE specification is in error.

ILLEGAL SORT FILE OPERATOR

ILLEGAL SORT INPUT-OUTPUT SPECIFICATION

Printed if a technique was applied to a sort-file. This is a warning only message and the TECHNIQUE is ignored.

ILLEGAL STATEMENT XXXXX

The word NEXT is not followed by SENTENCE, or the phrase TO PROCEED TO is missing in an ALTER statement.

ILLEGAL STATEMENT GROUP XXXXX

A. The word SENTENCE does not follow NEXT, or is misspelled.
B. A conditional statement must be followed by ELSE, OTHERWISE, or a PERIOD.

ILLEGAL SUBSCRIPT COPY OPERAND

The COPY statement illegally refers to a subscripted data-name.

ILLEGAL SUBSCRIPT MOVE OPERAND

A. The MOVE CORRESPONDING illegally refers to a subscripted data-name.
B. OPTION 3 of the MOVE must not have subscripted operands.

ILLEGAL SUBSCRIPT OPERAND

Either the sending item or the receiving item requires a subscript.

ILLEGAL SYNTAX DECLARATION XXXXX

A RENAMES entry is not allowed as part of a diagnostic statement.

ILLEGAL TYPE XXXXX

The qualifier is not a group item or a record-name.

ILLEGAL USAGE OPERAND SIZE

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

ILLEGAL VALUE

This message is due to:

A. The VALUE given is not within the allowable range or is improper, or
B. A VALUE may not be given for a data-name in the FILE SECTION or a VALUE stated for a label record field is not allowed.

ILLEGAL VALUE ASSIGNMENT XXXXX

This occurs in the DUMP "label: data-name", but the data-name is not an elementary item or not numeric.

ILLEGAL VALUE DECLARATION

The SAVE-FACTOR value is illegal.

ILLEGAL VALUE NAME XXXXX

The CLASS of the data-name XXXXX does not permit the stated VALUE.

ILLEGAL WRITE NAME XXXXX

The WRITE statement must refer to an 01 level record-name appearing in the FILE SECTION with an FD description, not to an SD record-name, not a record-name appearing in the WORKING-STORAGE or CONSTANT SECTIONS.

INPUT-OUTPUT MISSING XXXXX

The word INPUT or OUTPUT is missing on the USE statement XXXXX.

INTEGER CONDITIONAL OPERAND ERROR

This message indicates the following:

A. An ALPHANUMERIC item is not allowed.
B. A comparison of NUMERIC with a non-numeric item, and the NUMERIC item is not unsigned, an integer, or of DISPLAY usage.

INTO MISSING

LABEL MISSING XXXXX

The paragraph following a USE statement does not contain a label.

LEVEL ERROR

This message is used upon the occurrence of one of the following conditions:

A. A level-number larger than 49 in a record description.
B. A 77 level-number in the FILE SECTION.
C. A 66 level-number that is not associated with a RENAMES entry.
D. A 77 level-number that appears after the series of 77 level-numbers has been broken.

LEVEL NOT RIGHT

Compiler malfunction. Please report details.

LIBRARY COPY SELECTED

Copy contains nested copy.

LIBRARY nnnn

LIBRARY READ ERROR

Error occurred in READ FROM LIBRARY.

LITERAL OPERATOR LITERAL ERROR

The statement indicates a literal is compared with a literal.

LITERAL SIZE ILLEGAL XXXXX

LITERAL XXXXX CHARACTERS

A non-numeric literal longer than 120 characters. Length XXX.

LITERAL SPELLING ILLEGAL XXXXX

LITERAL SYNTAX PARENTHESIS

LITERAL TRUNCATION

The literal is stated out of the range of the item.

LITERAL VALUE NAME XXXXX

The value of XXXXX is not proper in the MOVE statement, or it is not a proper item for an arithmetic statement.

MEMORY SIZE NNNN

Requested information.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

MISSING ARITHMETIC OPERAND XXXXX

One of the following conditions is present:

A. No receiving field following the TO.
B. Only one operand shown.
C. The word XXXXX is not proper in the statement.

MISSING ASSIGNMENT

The SELECT clause should be followed by an ASSIGN clause.

MISSING ASSIGNMENT OPERATOR

The FROM, n, or EQUALS is missing in the COMPUTE statement.

MISSING AT END READ STATEMENT

Either the first READ statement for a file must have an AT END explicitly given and no other reads in program have the explicit AT END, or else every READ statement for the file in entire program must have an explicit AT END statement.

MISSING CONDITIONAL OPERAND XXXXX

The XXXXX is not a conditional operand.

MISSING CONDITIONAL OPERATOR XXXXX

A relational operator should appear prior to XXXXX.

MISSING DECLARATION SECTION

The program does not contain a section referred to by the USE statement in the DECLARATIVES, or a misspelling caused it to appear to be missing.

MISSING DIVISION

The heading for a division is missing, or is misspelled.

MISSING END DEC ILLEGAL PROCEDURE DIVISION

End declarative terminator not present.

MISSING FILE NAME

A file-name must follow the words INPUT or OUTPUT in the OPEN statement.

MISSING FILE SECTION SPECIFICATION XXXXX

The heading XXXXX appears instead of FILE SECTION.

MISSING FILE SECTION

MISSING FILE SPECIFICATION

A reference to BLOCK-COUNT, RECORD-COUNT, or REEL-NUMBER outside of a USE procedure is not qualified by the file-name.

MISSING GO TO PROCEDURE

The ALTER refers to other than a GO TO paragraph.

MISSING INPUT-OUTPUT OPERAND

The word INPUT or OUTPUT is omitted from the USE statement.

MISSING INPUT-OUTPUT SPECIFICATION

Invalid key clause missing in disk WRITE statement, or the verb OPEN is not followed by INPUT, OUTPUT, I-O or INPUT-OUTPUT.

MISSING LABEL

A label must identify the first paragraph of a section.

MISSING LEFT PARENTHESIS XXXXX

A left parenthesis is omitted:

A. Instead of the word XXXXX, or
B. Around the argument for an intrinsic function, or
C. Around a diagnostic statement list.

MISSING OPERATOR XXXXX

The word BEFORE or AFTER is not present in the USE statement.

MISSING PARENTHESIS XXXXX

MISSING PERIOD

A required period is missing.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

MISSING PERIOD XXXXX

A period is expected instead of the name or the symbol shown by XXXXX, or the diagnostic statement does not end with a period.

MISSING PROCEDURE DIVISION

The heading PROCEDURE DIVISION is omitted.

MISSING PROGRAM IDENT

The non-numeric literal of the Program-Id inside " " is missing.

MISSING QUALIFICATION

The word In or OF is omitted from a qualification.

MISSING QUALIFICATION XXXXX

The word XXXXX requires IN or OF as part of qualification.

MISSING QUALIFICATION NAME XXXXX

MISSING QUALIFICATION NAME

The necessary qualification is missing or the word used as a qualifier cannot be found in the program.

MISSING RECORD LEVEL

An 01 level record-name entry is omitted, or an 01 level record-name does not begin a record description following the 77 level entries.

MISSING RECORD SIZE

A data name has appeared where the compiler expected to find a level-number.

MISSING RIGHT PARENTHESIS

The terminating parenthesis following a synonym is missing.

MISSING RIGHT PARENTHESIS XXXXX

A right parenthesis should appear instead of XXXXX:

A. At the end of an arithmetic expression in a COMPUTE statement.
B. In a conditional clause.
C. Terminating the list in a diagnostic statement.

MISSING SECTION

The word SECTION is missing from a DATA DIVISION heading.

MISSING SIZE DEPENDING DECLARATION

MISSING SIZE DEPENDING DECLARATION FILE XXXXX

MISSING SIZE DEPENDING DECLARATION RECORD XXXXX

A file declared as TECHNIQUE-B or TECHNIQUE-C does not have a variable-length data record.

MISSING SIZE SPECIFICATION

The SIZE is not specified for an elementary item, or SIZE is not specified for a group item with a VALUE clause. Level-number is missing on item following a group item.

MISSING STOP RUN STATEMENT

A STOP RUN statement does not appear in the program, or it has been skipped due to a NOTE.

MISSING SYNTAX OPERATOR XXXXX

The condition stated in a DUMP diagnostic statement does not contain a colon after XXXXX.

MISSING XXXXX READ STATEMENT

MISSING XXXXX WRITE STATEMENT

MONITOR STATEMENT MISSING

The statement OPEN OUTPUT DIAGNOSTIC or CLOSE DIAGNOSTIC appears, but the compiler did not find a DUMP or MONITOR.

MOVE SYNTAX ERROR

The word following the CORRESPONDING in the MOVE statement is not proper, or a non-unique data name (possibly a synonym) used with the CORRESPONDING option, or the word TO is missing in a MOVE statement.

MOVE TRUNCATION

Due to differences in the description of the items in the MOVE statement, truncation of digits will occur.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

NEWTAPE nnnn

NO ELEMENT. ITEMS IN MOVE GROUP

A MOVE CORRESPONDING statement is given for which there are no corresponding elementary items. The level hierarchy of the data-names referenced by the CORRESPONDING option must match.

NOT RIGHT LEVEL

Compiler malfunction. Please report details.

NO. SEGS. NNN

Information on number of segments.

NOT SELECTED SORT TAPES

The SD sort-file description file-name is not the subject of a SELECT file-name ASSIGN TO n SORT-TAPES in the ENVIRONMENT DIVISION. n is an integer from 3 to 8.

NOT FILE NAME XXXXX

The symbol shown in a READ statement is not a file-name.

NOT RECORD NAME XXXXX

NOT RECORD DECLARATION XXXXX

The symbol XXXXX shown as an 01 record-name does not appear in the DATA RECORDS clause in the file description entry.

OPERAND XXXXX NOT INTEGER

The data-name XXXXX is not an integer quantity for the PERFORM statement to execute integer TIMES.

OPERAND RIGHT FILE RECORD XXXXX

OPERAND SIZE ERROR

A DISPLAY statement refers to data-name(s) whose total SIZE is greater than 176 characters.

OPTNL DECLARATION ERROR

Disk file may not be OPTIONAL.

PICTURE ERROR

The PICTURE specification is not proper, or the number of symbols in a PICTURE exceeds 30.

PICTURE PARENTHESIS USAGE ERROR

The number within parentheses specifying repetition is not an integer.

PICTURE SIZE ILLEGAL XXXXX

POSSIBLE ERROR RECORD SIZE

The RELEASE statement uses the FROM option, but the size of the two record areas is not the same. The Compiler will use the shorter length of the two.

POSSIBLE MOVE CLASS ERROR

Items are moved to a different CLASS item.

PROCEDURE MISSING XXXXX

The USE statement refers to a label that is not a part of the DECLARATIVES.

PROCEDURE SIZE ERROR

The generated code for this procedure exceeds 1023 words in length. An additional dummy label should be added to the procedure.

PROCEDURE XXXXX SIZE XXXXX

Information on procedure size.

PRT NNN

Requested PRT number.

PRT SIZE NNN

PRT SIZE ERROR

This message is due to:

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

A. The program Reference Table has exceeded 511 words. Reduce the number of 01 levels and COMP-1 items (in DATA DIVISION), or
B. The Program Reference Table has exceeded 1023 words. Reduce the number of labels used in the program (in PROCEDURE DIVISION).

QUOTE MISSING LITERAL GREATER THAN 120 CHARACTERS

READ STATEMENT SYNTAX ERROR

In a READ statement, the word END is missing, or SENTENCE in the clause AT END GO TO NEXT SENTENCE is missing.

RECORD SIZE ERROR

The record size of a sort-file is not equal to the record size of the input or output file of the sort. The program is not compiled if the record size is less than the sort record size.

RECORD SIZE SELECTED; SORT VECTOR SIZE XXXXX

REDEFINE ERROR

The operand of a REDEFINES clause is illegal.

REDEFINE SIZE ERROR

The area being redefined does not equal the size of the new description.

REMOTE IMPROPER FOR NORMAL SYS

RIGHT PARENTHESIS MISSING

The synonym entry requires a terminating right parenthesis.

RIGHT QUOTE MISSING

SEQUENCE ERROR

The sequence number appearing in card columns 1 thru 6 is not greater than the number of the preceding card. The message is printed but compilation is unharmed.

SEQUENCE ERROR nnnn

SEQUENCE NUMBER TRUNCATION XXXXX

SIZE DECLARATION ERROR

The declared size of the item, and the size shown by a PICTURE do not equal.

SIZE ERROR STATEMENT XXXXX

The word SIZE or ERROR is missing from an ON SIZE ERROR clause, or the statement containing the word or symbol shown by XXXXX has caused a segment of code to exceed 1023 words in length. Additional labels should be added to reduce the SIZE of the segment.

SIZE ERROR WRITE STATEMENT

SIZE ILLEGAL ACT. KEY

SIZE ILLEGAL SIZE DEPENDING OPERAND

The variable size item in this statement has a SIZE DEPENDING operand with a size greater than 11.

SIZE ILLEGAL SIZE DEPENDING OPERAND FILE XXXXX

The SIZE DEPENDING data-name referred to by the READ statement contains more than 11 decimals digits.

SIZE ILLEGAL SIZE DEPENDING OPERAND RECORD XXXXX

The SIZE DEPENDING data-name referred to by the WRITE statement contains more than 11 decimal digits.

SIZE SPECIFICATION ERROR

An item declared as CMP-1 or COMPUTATIONAL-1 cannot exceed 11 decimal digits in length.

SORT MEMORY SIZE

Amount of memory used for a sort if an insufficient amount were specified.

SORT USAGE ERROR XXXXX

SORT VECTOR SIZE XXXXX

RECORD SIZE SELECTED XXXXX

SORT statement information. Provided only when SPEC appears in the $ card and MEMORY SIZE is below the minimum (two times record size) for the sort.

PRINTED IN U.S. AMERICA 8/70 1031986

STATEMENT GROUP SIZE ERROR

The group size is greater than 1023 words.

STATEMENT TRUNCATION

Too many items in DUMP/MONITOR.

SUBSCRIPT SPECIFICATION ERROR

An item declared as CMP-1 or COMPUTATIONAL-1 may not be subscripted.

SUBSCRIPT TRUNCATION

May not monitor an item with more than 10 subscripts.

SUBTR. NO ELEMENT. ITEMS

SYNTAX ERROR

In the IDENTIFICATION DIVISION, the word DIVISION is missing following IDENTIFICATION.

Any of the following errors within the ENVIRONMENT DIVISION will cause this message:

A. Omission of the word DIVISION.
B. Omission of a period.
C. Statement form is wrong.
D. Invalid hardware-name.
E. No SOURCE-COMPUTER statement. (Warning only)
F. MODS used instead of MOD in DISK SIZE.

Any of the following errors within the DATA DIVISION will cause this message:

A. A non-numeric literal is specified when a numeric literal is needed.
B. A numeric literal is specified when a non-numeric literal is needed.
C. The VALUE clause is missing.
D. The format of the literal does not match the item description.
E. An OCCURS clause, with a variable number of times, omitting the DEPENDING ON specification.
F. The word SECTION is missing from the headers.
G. An FD or an SD is used incorrectly.
H. A reserved word is used incorrectly.
I. A missing data-name.
J. A figurative constant is used incorrectly.
K. An incorrect declaration. (i.e., no level-number)
L. The item following SIZE (or SZ) is not numeric.
M. The item following OCCURS is not numeric.

Any of the following errors within the PROCEDURE DIVISION will cause this message:

A. An illegal operator in an EXAMINE statement.
B. An incorrect record-name (other than one defined by an SD) included in a RELEASE statement.
C. The END-OF-JOB Card is misplaced in the source program.
D. Reserved word being EXAMINED.
E. READ or WRITE on an SD file.
F. The preceding statement is incomplete.
G. Tape opened I-O.
H. A GO TO specifies a data-name.

SYNTAX ERROR XXXXX

The XXXXX is in error.

SYNTAX ERROR DIVISION MISSING

The word DIVISION is missing or misspelled following PROCEDURE in the heading.

SYNTAX ERROR FILE DECLARATION XXXXX

The file-name used is a reserved word or has previously been used.

SYNTAX ERROR GO TO ERROR

Word after GO is not TO.

SYNTAX ERROR GO TO STATEMENT

The word TO does not follow GO TO statement, or the DEPENDING ON clause is missing.

SYNTAX ERROR ILLEGAL SPELLING

SYNTAX ERROR LIBRARY MISSING

LIBRARY missing following FROM.

SYNTAX ERROR MISSING FILE NAME

In a CLOSE state, the word following CLOSE is not a file-name.

SYNTAX ERROR MISSING LABEL

A required label is missing in a statement, such as ALTER label, TO PROCEED TO label, or GO TO label.

PRINTED IN U.S. AMERICA 8/70

1031986

SYNTAX ERROR MISSING LITERAL

The STOP statement is not followed by the reserved word RUN or by a literal.

SYNTAX ERROR MISSING PERIOD

A required period is missing.

SYNTAX ERROR MISSING PERIOD XXXXX>

The required period following XXXXX is missing.

SYNTAX ERROR MISSING QUALIFICATION XXXXX

XXXXX is not a valid qualifier, or is MULTIPLY defined without qualifications.

SYNTAX ERROR MISSING VERB

SYNTAX ERROR MOVE STATEMENT

SYNTAX ERROR NAME -DATA-

SYNTAX ERROR NAME -REMOTE-

SYNTAX ERROR PERFORM STATEMENT

SYNTAX ERROR READ STATEMENT

SYNTAX ERROR SORT STATEMENT 1

The SORT statement is not the first statement of the paragraph, or an attempt has been made to MONITOR a paragraph which contains a SORT statement.

SYNTAX ERROR SORT STATEMENT 2

The name of the sort-file cannot be located in the program. This is probably due to misspelling.

SYNTAX ERROR SORT STATEMENT 3

The word following SORT is not a file-name.

SYNTAX ERROR SORT STATEMENT 4

The file-name given following SORT is an FD file description instead of an SD sort-file description.

SYNTAX ERROR SORT STATEMENT 5

The wrong word appears following the sort file-name. Normally, this word is ON, or ASCENDING, or DESCENDING.

SYNTAX ERROR SORT STATEMENT 6

The word following ON is incorrect, possibly misspelled.

SYNTAX ERROR SORT STATEMENT 7

The ordering of the SORT statement into ASCENDING or DESCENDING sequence is not specified.

SYNTAX ERROR SORT STATEMENT 8

There are more than 25 keys used in the SORT statement ordering.

SYNTAX ERROR SORT STATEMENT 9

There are more than 25 keys used in the SORT statement ordering.

SYNTAX ERROR SORT STATEMENT 10

The word ASCENDING or DESCENDING is either missing or misspelled.

SYNTAX ERROR SORT STATEMENT 11

One of the key names given the ordering key cannot be located in the program.

SYNTAX ERROR SORT STATEMENT 12

The SORT statement KEY data-name has a USAGE that is neither DISPLAY nor COMPUTATIONAL. This is due to a system failure of some type; either the Master Control Program, the COBOL Compiler, or the hardware caused the error.

SYNTAX ERROR SORT STATEMENT 13

The CLASS of the SORT statement KEY data-name is not correct. This is due to a system failure within the Master Control Program, the COBOL Compiler, or the hardware.

PRINTED IN U.S. AMERICA 8/70

1031986

SYNTAX ERROR SORT STATEMENT 14

The SIGN of the SORT statement KEY data-name is not correct. This is due to a system failure within the Master Control Program, the COBOL Compiler, or the hardware.

SYNTAX ERROR SORT STATEMENT 15

The SORT statement KEY data-name requires subscripting that is not present.

SYNTAX ERROR SORT STATEMENT 16

The subscript for the data-name in the SORT statement KEY is not an unsigned integer quantity, and is illegal.

SYNTAX ERROR SORT STATEMENT 17

The closing parenthesis following a subscript list for a SORT statement KEY data-name is missing.

SYNTAX ERROR SORT STATEMENT 19

The word following a SORT statement KEY data-name cannot be located in the program, possibly due to misspelling.

SYNTAX ERROR SORT STATEMENT 20

The word following a sort KEY is not found in the dictionary due to misspelling, etc.

SYNTAX ERROR SORT STATEMENT 21

A reserved word, or a symbol such as comma or right parenthesis, was expected after one of the SORT statement KEY data-names, but is not present, or is unable to be identified due to misspelling, etc.

SYNTAX ERROR SORT STATEMENT 22

The word following INPUT cannot be properly identified.

SYNTAX ERROR SORT STATEMENT 23

The word following INPUT is not PROCEDURE.

SYNTAX ERROR SORT STATEMENT 24

The SORT statement does not contain an INPUT PROCEDURE, therefore, the USING file-name must be present, but USING cannot be located probably due to a spelling error.

SYNTAX ERROR SORT STATEMENT 25

The file-name following the USING either is not a file-name, or is misspelled.

SYNTAX ERROR SORT STATEMENT 26

The file-name following the USING cannot be identified as a file-name.

SYNTAX ERROR SORT STATEMENT 27

The file-name following USING has an SD sort-file description entry instead of an FD file description.

SYNTAX ERROR SORT STATEMENT 28

The word following the USING file-name clause, or the INPUT PROCEDURE, cannot be identified.

SYNTAX ERROR SORT STATEMENT 29

The word following OUTPUT is not PROCEDURE.

SYNTAX ERROR SORT STATEMENT 30

An OUTPUT PROCEDURE is not specified in the SORT statement, therefore, GIVING file-name should be present.

SYNTAX ERROR SORT STATEMENT 31

The data-name following GIVING cannot be identified in the program.

SYNTAX ERROR SORT STATEMENT 32

The data-name following GIVING is not a file-name.

SYNTAX ERROR SORT STATEMENT 33

The output file-name following GIVING is described with an SD sort-file description instead of an FD file description.

SYNTAX ERROR SORT STATEMENT 34

The period is missing following the SORT statement. No other statement is permitted within the same sentence, or paragraph, with the SORT statement.

SYNTAX ERROR SORT STATEMENT 35

The period terminating the SORT statement sentence is missing.

PRINTED IN U.S. AMERICA 8/70 1031986

SYNTAX ERROR SORT STATEMENT 36

The INPUT PROCEDURE and the OUTPUT PROCEDURE both refer to the same set of procedures. This is illegal.

SYNTAX ERROR SORT STATEMENT 37

The SORT statement is attempting to use PRT locations in the second half of the PRT.

SYNTAX ERROR SORT STATEMENT 38

A warning message indicating more than one SORT statement is using the same SD file as scratch tapes.

SYNTAX ERROR SORT STATEMENT 39

Sort key is not in the sort record.

SYNTAX ERROR SORT STATEMENT 41

One or more of the sort keys exceed 63 characters.

SYNTAX ERROR XXXXX STATEMENT

SYNTAX ERROR WRITE STATEMENT

Illegal advancing operand.

SYNTAX ERROR VERB SYNTAX ERROR

SYNTAX TYPE OPERAND

Improper use of reserved word in EXAMINE statement, or the literal intended for the SEARCH is not bounded by quotes.

TAPE-IN nnnnn

THIS PERFORM OPTION DELETED FROM TSS

TO MISSING XXXXX

XXXXX appears after EQUAL instead of TO.

TOTAL SEG. SIZE NNNNN

Requested information.

TYPE ILLEGAL ACT. KEY

UNIDENTIFIED ARITHMETIC NAME XXXXX

The data-name XXXXX cannot be located in the program.

UNIDENTIFIED ARITHMETIC OPERAND XXXXX

The data-name or symbol cannot be located in the program, probably due to spelling errors.

UNIDENTIFIED COPY OPERAND

The data-name following COPY cannot be located in the program thus far due to spelling errors or a forward reference.

UNIDENTIFIED HARDWARE

The hardware-name used is not permitted in the Compiler.

UNIDENTIFIED LIBRARY NAME XXXXX

Name cannot be located on library.

UNIDENTIFIED NAME

The name given cannot be located in the program.

UNIDENTIFIED NAME XXXXX

The data-name or label XXXXX cannot be located in the program, or the compiler is looking for a reserved word. RECORD or CHARACTER may be misspelled in BLOCK CONTAINS.

UNIDENTIFIED OPERAND XXXXX

The data-name given in a forward reference is not in the DATA DIVISION.

UNIDENTIFIED RECORD XXXXX

A record-name XXXXX defined by an 01 level entry is not given in the DATA RECORDS clause, or a record-name XXXXX appearing in the DATA RECORDS clause does not appear as an 01 level entry.

UNIDENTIFIED REDEFINE OPERAND

Operand does not appear in prior description.

PRINTED IN U.S. AMERICA 8/70 1031986

UNIDENTIFIED VERB XXXXX

The verb beginning the statement cannot be identified by the Compiler.

-UNTIL- DELETED FROM TSS

USAGE ERROR

This message is due to:

A. A COMPUTATIONAL usage has been declared for a file which is to a unit other than tape or drum. (If item is COMPUTATIONAL, then it will be a binary word), or
B. Usage not declared as DISPLAY, or COMPUTATIONAL, or CMP, or else omitted completely to imply DISPLAY, or
C. Usage must be DISPLAY for item in EXAMINE statement.

USAGE SPECIFICATION ERROR

VALUE NOT INTEGER XXXXX

The value stated in the diagnostic dump statement, as the condition when the statement is to be executed, is not an integer.

VALUE TYPE ERROR

The VALUE stated for a level 88 entry does not agree with the CLASS given for the conditional-variable.

-WHEN- DELETED FROM TSS

SECTION 5
DUMP DEBUGGING AIDS

DUMP DECODING AIDS

SALF	MSFF	T12F A10F	T11F A09F	T10F A08F	BASE	INDEX BITS	ADDRESSABLE AREA SIZE
OFF	-	-	-	-	R+	T (12 - 3) A (10 - 1)	(1,024)
ON	-	OFF	-	-	R+	T(11 - 3) A(9 - 1)	(512)
ON	OFF	ON	OFF	-	F+	T(10 - 3) A(8 - 1)	(256)
ON	ON	ON	OFF	-	(R+7)+ *	T(10 - 3) A(8 - 1)	(256)
ON	-	ON	ON	OFF	C+**	T (9 - 3) A(7 - 1)	(128)
ON	OFF	ON	ON	ON	F-	T(9 - 3) A(7 - 1)	(128)
ON	ON	ON	ON	ON	(R+7) -	T(9 - 3) A(7 - 1)	(128)

- Irrelevant setting
* Relative addressing using as the base, bits 16 thru 30 of the word Stored in the programs PRT at R+7.
** "C" relative coding is forced to "R" relative for the Store, Program and I/O Release Operators.

DESCRIPTOR FORMATS

Word Mode program descriptor (Spontaneous entry type) 740000FFFFFCCCCC

Word Mode program descriptor 75000000000CCCCC

Label descriptor 760000FFFFFCCCCC

Character Mode program descriptor 77000000000CCCCC

Data descriptor (Information not present) 40*WWWFFFFFCCCCC

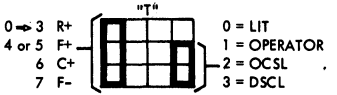
*— low order bit also W

Data descriptor (Information present) 50*WWWFFFFFCCCCC

*— low order bit also W

W = Word Count F = F register setting C = Core Address in reg.

BASE REGISTER AND SYLLABLE TYPE



PRINTED IN U.S. AMERICA REVISED 8/70

1031986

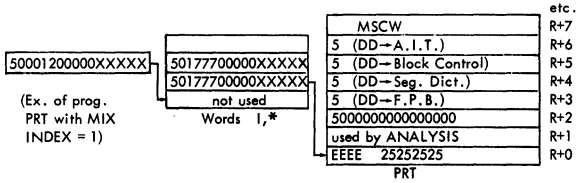
CELL DESIGNATION FOR ADDRESSES ONE TO ONE HUNDRED OCTALLY

This refers to common usage in "R" relative or absolute

01	
02	
03	
04	Work Area
05	
06	
07	Mark Stack Control Word
10	Initiate Control Word for P2 or I/O Descriptor Address
11	Address of word modified by program release operator
12	
13	
14	I/O 1 Result Descriptor
15	I/O 2 Result Descriptor
16	I/O 3 Result Descriptor
17	I/O 4 Result Descriptor
20	First cell read on from Halt/Load
21	
22	Time Interval Interrupt (CC103F)
23	I/O Busy Interrupt (CC104F)
24	Keyboard Request Interrupt (CC105F)
25	Printer 1 Finished Interrupt (CC106F)
26	Printer 2 Finished Interrupt (CC107F)
27	I/O 1 Finished Interrupt (CC108F)
30	I/O 2 Finished Interrupt (CC109F)
31	I/O 3 Finished Interrupt (CC110F)
32	I/O 4 Finished Interrupt (CC111F)
33	P2 Busy Interrupt (CC112F)
34	Inquiry Request Interrupt (CC113F)
35	Special Interrupt 1 (CC114F)
36	Disk File 1 Finished Interrupt (CC115F)
37	Disk File 2 Finished Interrupt (CC116F)
40	P2 Memory Parity Error (Pk-I01F)
41	P2 Invalid Address (Pk-I02F)
42	P2 Stack Overflow (Pk-I03F)
44	P2 Communicate (Pk-BCD 4)
45	P2 Program Release (Pk-BCD 5)
46	P2 Continuity Bit (Pk-BCD 6)
47	P2 Presence Bit (Pk-BCD 7)
50	P2 Flag Bit (Pk-BCD 8)
51	P2 Invalid Index (Pk-BCD 9)
52	P2 Exponent Underflow (Pk-BCD 10)
53	P2 Exponent Overflow (Pk-BCD 11)
54	P2 Integer Overflow (Pk-BCD 12)
55	P2 Divide by Zero (Pk-BCD 13)
56	Not Used
57	Not Used
60	P1 Memory Parity Error (Pk-I01F)
61	P1 Invalid Address (Pk-I02F)
62	P1 Stack Overflow (Pk-I03F)
63	Not Used
64	P1 Communicate (Pk-BCD 4)
65	P1 Program Release (Pk-BCD 5)

66	P1 Continuity	(Pk-BCD 6)
67	P1 Presence Bit	(Pk-BCD 7)
70	P1 Flag Bit	(Pk-BCD 8)
71	P1 Invalid Index	(Pk-BCD 9)
72	P1 Exponent Underflow	(Pk-BCD 10)
73	P1 Exponent Overflow	(Pk-BCD 11)
74	P1 Integer Overflow	(Pk-BCD 12)
75	P1 Divide by zero	(Pk-BCD 13)
76	Not Used	
77	Not Used	
100	Base of MCP Stack	

ARRAY PRT [*, *]



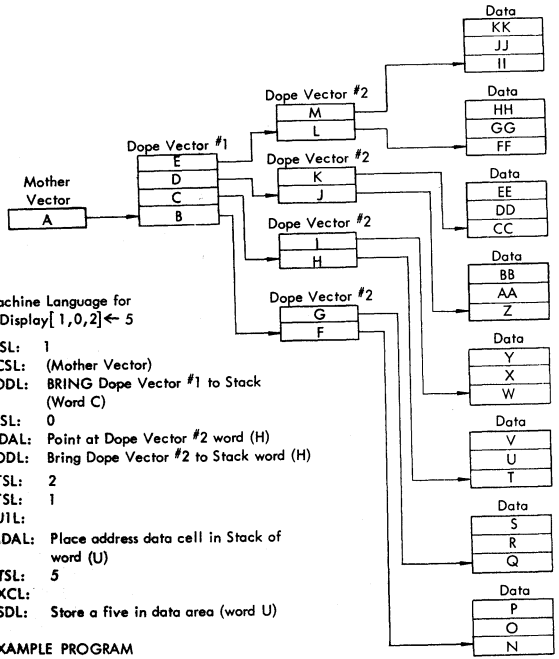
CONTENTS OF THE FIRST 25 (OCTAL) PRT LOCATIONS OF AN ALGOL OBJECT PROGRAM

R+0	"EEEEEEEE"	Used by DF MCP to denote beginning of PRT.
R+1		Used by ANALYSIS for branch to non-present label.
R+2	5000.....0	"Memory" for normal state.
R+3	FPB	Descriptor pointing to FILE BLOCK (FPB).
R+4	SD	Descriptor pointing to SEGMENT DICTIONARY (SD).
R+5	BC	Descriptor pointing to BLOCK CONTROL intrinsics.
R+6	AIT	Descriptor pointing to ARRAY INFORMATION TABLE.
R+7	MSCW	Mark Stack Control Word
R+10	INCW	Initiate Control Word
R+11	COM/PRL	a) Stores address of word modified by program release operator. b) Temporary storage for word stored by communicate operator.
R+12		Data descriptor referencing base of PRT. (The F-register field contains location of stack bottom.)
R+13	SIZEERROR/ OWN ARRAY	Used to handle ON SIZE ERROR clause in COBOL. Descriptor pointing to (OAT) OWN ARRAY TABLE in ALGOL.
R+14	ALGOL WRITE/ COBOL FCR	Program descriptor pointing to write intrinsics for ALGOL, FCR for COBOL.
R+15	ALGOL READ	Program descriptor for read intrinsics in ALGOL.
R+16	ALGOL SELECT	Program descriptor pointing to ALGOL SELECT.
R+17	0	ZERO
R+20	BLOCKCTR	Block level counter (starts at 1 with outermost block of symbolic programs).
R+21	JUNK	(Temporary storage location for use by software.)
R+22	EXITR	Character mode descriptor -- references the first syllable of the program, i.e., the outermost block which is generated by the compiler.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

R+23 R+24	LISTRTN	Used to obtain the next element of a list. Program descriptor of block number 2 (i.e., the block which corresponds to the outermost block of the symbolic program).
R+25	ERROR COUNT	Storage location used by compiler to store Syntax error count. First PRT LOCATION ASSIGNED BY COMPILER.

ARRAY FORMAT



Machine Language for
Display[1,0,2]← 5

```
LTSL:  1
DCSL:  (Mother Vector)
LODL:  BRING Dope Vector #1 to Stack
       (Word C)
LTSL:  0
MDAL:  Point at Dope Vector #2 word (H)
LODL:  Bring Dope Vector #2 to Stack word (H)
LTSL:  2
LTSL:  1
SUIL:
MDAL:  Place address data cell in Stack of
       word (U)
LTSL:  5
EXCL:
BSDL:  Store a five in data area (word U)
```

EXAMPLE PROGRAM

```
BEGIN
   Array Display [0:3, 0:1, 1:3] ; (Mother Vector
   generated here and placed in PRT)
   Display [1,0,2] ← 5;
END
```

The end result of this program is to place a "5" in word U of array display.

METHOD FOR DECLARING ARRAY SPACE WITH DF MCP

The call on the DF MCP to declare array space is nearly identical to the call made when using the MD MCP. That is, the same parameters are required in the stack (with the exception that a different literal value is used to specify the type of storage); however, an operand call on a block control intrinsic program descriptor is used rather than a communicate operator, when the DF MCP is called.

Explicitly, the following parameters are required in the stack:

1. MSCW.
2. Descriptors pointing to the array descriptors for each array being declared.
3. Sizes of the array dimensions.
4. Number of dimensions.
5. TYPE of storage.

With these parameters in the stack an operand call on the block intrinsic program descriptor will cause the array space setup.

The values for TYPE are defined as follows:

0 = Regular array space (overlayable).
1 = SAVE array space (non-overlayable).
2 = OWN array space.
3 = SAVE and OWN array space.

INTERROGATION AIDS USING AN I/O CHANNEL

An I/O Channel can be forced to fetch the address of an I/O Descriptor from any address in Core Memory. Normally, when initiating an I/O operation, the I/O will interrogate cell 10 for the address. The ability to fetch the address of the descriptor is particularly handy when performing magnetic tape operations, and a rewind is required. The following described procedure can be used, however, with any I/O operation.

When initiating a local I/O operation, the logic at SC=0 causes D17F to be set along with address 10_0 being jammed into [D15⇒1] . D17F causes a fetch then from 10_0 for the address of the I/O descriptor and then commences to get the descriptor and cause appropriate action. If the [D15⇒1] is set to point at another cell in core memory which contains an address of an I/O descriptor and D17F is manually set, the same fetch of that address of the descriptor begins and the subsequent identical I/O action on the descriptor.

The following describes how this function can be used in Tape Operation:

When Writing information on a magnetic tape using recycle and local switches, D26F can be manually set at any time and the tape unit will do a rewind. However, if the I/O were doing a Read operation, the setting of D26F would cause backward reads while it is set, and then resume forward Reads when it is released.

By placing a tape "Rewind" descriptor somewhere in Core Memory and then placing its address in cell 0_0, a rewind can be initiated at any time by simply setting D17F manually.

PRINTED IN U.S. AMERICA 10-67

1031986

USE OF MEMORY LOAD SWITCH IN TROUBLESHOOTING

The Memory Load Switch can be used in three ways:

1. To load specific information into a designated cell in memory.
   a. Master Clear
   b. Inhibit Time Interval Interrupt
   c. Set CL2F on the CC Panel
   d. Throw MEMORY LOAD Switch to Upward Position
   e. Place into the B Register the contents wanted in the cell and into the S Register the address of the cell that the contents of the B Register is written into.
   f. Depress the MEMORY LOAD BUTTON ONCE
2. To load specific information into all cells above a certain address:
   a. Do operation a, b, c, d and e as written above.
   b. Throw INHIBIT INTERRUPTS switch to the upward position.
   c. Depress and hold the HALT flip-flop button. This will write the word in the B Register into the address in the S Register and all addresses above this address to the top cell in memory.
3. To locate a cell in memory where there is a Parity Error:
   a. Do operation a, b, c and d as in Section 1.
   b. Depress the "2" bit button in the E Register. This will allow the Processor to stop the S Register pointing the error cell plus 1. The word in the A Register is the information from the error cell.

TYPICAL STACK STATUS AT TIME OF INTERRUPT

WORD MODE: R+10 INCW PRT
IRCW
ICW
A reg. (AROF)
B reg. (BROF)
Local Variable
Local Variable
Local Variable
RCW
Parameters
Parameters
Parameters
MSCW
data
data
data
MSCW
Stack base cell

CHARACTER MODE: R+10 INCW PRT
IRCW
ICW
ILCW
B reg. (BROF)
A reg. (AROF)
Loop Cont. Wd
Loop Cont. Wd
RCW
Local Variable
Local Variable
Parameter
Parameter
MSCW
data
data
MSCW
Stack base cell

MEMORY LINKS

Memory link words are used to keep track of the organization and classifications assigned to core memory. Two types of memory links are utilized: (1) link for available storage and (2) link for in-use storage. There is a link preceding all areas of core memory. Three variables in the MCP PRT reference the memory links:

(1) INTEGER AVAIL_____; AVAIL contains the address of the stopper for available storage links. Its value is the highest available address-1.

(2) INTEGER MSTART_____; MSTART contains the address of the first area of storage after end of ESP program.

(3) INTEGER MEND_____; MEND points to the last storage link in memory.

The following formats are used for memory links:

LINK FOR AVAILABLE AREA (3 WORDS - AVAIL BIT = 1)

WORD 1

2/3	X	X	X	X	X	a	a	a	a	a	a	a	a	a	a	a	a
[0:1] = 0 FLAG						ADDR. OF 1ST WD IN LINK - PREVIOUS AREA						ADDR. OF 1ST WD IN LINK - NEXT AREA					
[1:1] = 1 AVAIL BIT																	
[2:16] = NA																	

WORD 2

0	0	0	0	0	0	W	W	W	W	W	a	a	a	a	a
[0:18] = 0 ZEROS REQUIRED BY LLL OPERATOR						SIZE OF THIS AVAILABLE AREA					ADDR. OF 2ND WD LINK - NEXT AVAILABLE AREA				

WORD 3

0	0	0	0	0	0	0	0	0	0	0	a	a	a	a	a
[0:33] = 0 NOT USED											ADDR. OF 2ND WD LINK - PREV. AVAILABLE AREA				

LINK FOR IN-USE AREA (2 WORDS - AVAIL BIT = 0)

WORD 1

0/1	T	T	M	M	X	a	a	a	a	a	a	a	a	a	a	a	a
[0:1] = 0 FLAG						ADDR. OF 1ST WD IN LINK - PREVIOUS AREA						ADDR. OF 1ST WD IN LINK - NEXT AREA					
[1:1] = 0 AVAIL BIT																	
[2:1] = 1 SAVE AREA																	

[3:6] = TYPE OF AREA (00 = MCP, 1 = PROGRAM, 2 = DATA, 3 = I/O BUFFER, 4 = ALGOL FIB, 5 = FILL WITH INQUIRY BUFFER, 6 = COBOL FILE, 7 = INTRINSIC SEG., 8 = HEADER)

[9:6] = MIX INDEX OF PROGRAM USING AREA.

WORD 2

S	S	S	S	S	S	S	S	S	S	S	a	a	a	a	a
[0:33] = PROGRAM SEGMENT SIZE											ADDRESS OF				

[33:15] = IF DATA, ADDRESS OF ARRAY DESCRIPTOR.
IF OBJECT PROG., SEGMENT NUMBER.
IF BUFFER AREA, TOP I/O DESCRIPTOR.
IF MCP PROG. SEG., PRT ADDRESS.
IF INTRINSIC, SEGMENT NUMBER (OCTAL)

[8:10] = IF INTRINSIC, INDEX INTO ARRAY INTRNSC (INTRINSIC NUMBER)

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

CORE MEMORY AT HALT-LOAD TIME MODULES 0, 1, 3 AND 4 ON LINE

MODULE 0

00000	1	0	1	0	0	0	4	7	7	7	5	0	3	7	2	0
00001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
03720	2	0	0	0	0	0	0	0	0	0	0	1	7	7	7	6
03721	0	0	0	0	0	0	1	4	0	5	4	3	0	0	0	1
03722	0	0	0	0	0	0	0	0	0	0	0	4	7	7	7	6

MODULE 1

17776	1	0	0	0	0	0	0	3	7	2	0	3	0	0	0	0
17777	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MODULE 3

30000	2	0	0	0	0	0	1	7	7	7	6	4	7	7	7	5
30001	0	0	0	0	0	0	1	7	7	7	4	4	7	7	7	6
30002	0	0	0	0	0	0	0	0	0	0	0	0	3	7	2	1

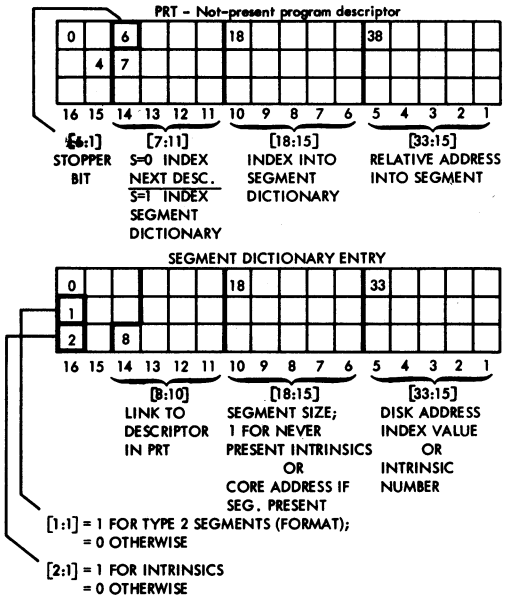
MODULE 4

47775	1	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0
47776	0	0	0	0	0	7	7	7	7	7	0	3	7	2	1
47777	0	0	0	0	0	0	0	0	0	0	3	0	0	0	1

SEGMENT DICTIONARY AND RELATED PRT CELLS AS CREATED BY THE COMPILER

Each program has a segment dictionary containing one entry for every program segment in the program, and one word for every intrinsic used. The first word is referenced as word zero; the entry for any particular segment is located in the word corresponding to that segment's number (e.g., the entry for segment 3 would be in the fourth word of the segment dictionary). Each segment dictionary entry may have one or more program descriptor, in the PRT, referencing a segment; some have none (e.g., fill segments). Segments with more than one program descriptor referencing the segment are linked (contain an index to next PD) until stopper bit is set indicating the last PD pertaining to the segment.

Format of PRT Not-present program descriptors and Segment Dictionary entries as created by the compilers is as follows:



COMPUTATION FOR THE DISK ADDRESS OF NON-PRESENT SEGMENTS

The index value found in [33:15] of a segment dictionary entry is a relative address into the programs code file on disk. Other pertinent information is found in the programs JAR:

JAR [8] = Number of disk segments in each row (value in octal).
JAR [10] thru JAR [29] = disk base address of each row (address in octal).
(e.g., JAR [10] for the 0 row, JAR [11] for the 1 row, etc.)

PRINTED IN U.S. AMERICA 10-67 1031986

To find the absolute disk address of a non-present program segment, take the (SEG DICT [33:15]) DIV (JAR [8]). This tells which disk row it is in. If this value is 0 then JAR [10] is used, if the value is 1 then JAR [11] is used, etc. Next, index this selected disk address by (SEG DICT [33:15]) MOD (JAR [8]); this is the address you seek.

Example:

SEG DICT [33:15] = 103 (octal)
JAR [8] = 100 (octal)
JAR [10] = 12345 (octal)
JAR [11] = 23602 (octal)
JAR [12] = 16651 (octal)

(SEG DICT [33:15]) DIV (JAR [8]) = 103 DIV 100 = 1
(SEG DICT [33:15]) MOD (JAR [8]) = 103 MOD 100 = 3

Therefore disk address of non-present program segment is:

23602 + 3 = 23605 (octal) - converted to decimal equals 10117.

THE FORMAT OF SEGMENT ZERO OF PROGRAMS

S[0] = Location of Segment Dictionary
S[1] = Size of Segment Dictionary
S[2] = Location of PRT
If S[2] < 0 then the job was compiled by COBOL
S[3] = Size of PRT
S[4] = Location of File Parameter Block
S[5] = Size of File Parameter Block
S[6] . [1:1] = 1 for new format segment 0, ELSE 0.
S[6] = Starting Segment Number.
S[7] . [33:15] = Number of Files.
S[7] . [18:15] = Core Requirement / 64.
S[15] = Disk address of LABEL EQUATION entries presented when program was compiled and applicable to all executions.
S[16] = Estimated processor time (from compilation)
S[17] = Estimated I/O time (from compilation)
S[18] = Priority (from compilation)
S[19] = COMMON VALUE (from compilation)
S[20] = Estimated core requirement (from compilation)
S[21] = Stack size (from compilation)

ARRAY JAR [* *] ____ JOBS ACTUALLY RUNNING

The table JAR for a given program can be located by indexing the Descriptor called JAR (located in the MCP's PRT) by the Mix index of the respective program, which will select a data descriptor, which in turn will point at the base of JAR for the respective program. If the PRT entry JAR plus a given index contains zeros, a program has not been assigned that Mix index value. The SELECTION routine will fill JAR from the SHEET when enough space is available to run a job.

J[0] = 1st Name (7 chrs) < 0 if a compiler
J[1] = 2nd Name (7 chrs) < 0 if job is being DS-ED

J[2] . [0:1] After SELECTION, if this program was compiled using COBOL = 1.
. [1:2] During SELECTION, as follows:
0 = normal
2 = job has been XS-ed
3 = job has been ES-ed
. [8:10] 0 = go job (from COMPILE-and-GO).
1 = compiler (COMPILE-and-GO).
2 = execute job.
3 = compiler (syntax check - set to 2 later).
4 = compiler (COMPILE-to-LIBRARY).
5 = run job.
99 = aborted job (from INITIALIZE).
1023 = syntax errors
. [18:15] Skeleton disk address (if JAR[2] . [8:10] = 1,2, or 4) for the skeleton SHEET for GO part.
. [33:15] Priority
J[3] . [8:10] Scheduled identification for this job.
. [33:15] Estimated processor time.
J[4] Estimated I/O time.
. [1:23] Starting date for the log (binary).
J[5] . [24:24] Starting time for log.
J[6] . [18:15] Size of log information in ESPDISK.
. [33:15] Location of the first record of the log information in ESPDISK. If J[2] . [8:40] = 0, then this is the compile log information.
J[7] Idle time.
J[8] Length of each row of the code file.
J[9] Number of rows.
J[10-29] Disk address for each row of the code file.

ARRAY BED [*]

The BED array, the SLEEP and COMPLEXSLEEP procedures are used to suspend the processing of an object program until a certain condition exists. Entries in BED consists of two words and is made through the SLEEP procedure. The last entry in BED is pointed to by JOBNUM. The BED is used by the NOTHINGTODO routine to restart jobs which have been temporarily suspended.

Entries made by the SLEEP routine (direct call on SLEEP)

Word 1 5 M M 0 0 0 F F F F F A A A A A
*(excluding low order bit)
[0:3] 5 Descriptor identification bits
[3:5] M MIX INDEX of suspended program
[8:10] 0 Size field not used
[18:15] F F field - Address of RCW of SLEEP procedure.
[33:15] A Address of word to be tested to determine if the necessary condition is satisfied.

Word 2 0 _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
[0:1] Flag bit (cannot be used for mask bit).
[1:47] MASK Contains 1's in bit positions which indicate when the needed condition is present. All other bits are set to zero.

Entries made by COMPLEXSLEEP calling on SLEEP

Word 1 0 M M 0 0 0 F F F F F 0 0 0 0 1
*(excluding low order bit)
[0:3] 0 Operand identification bits.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

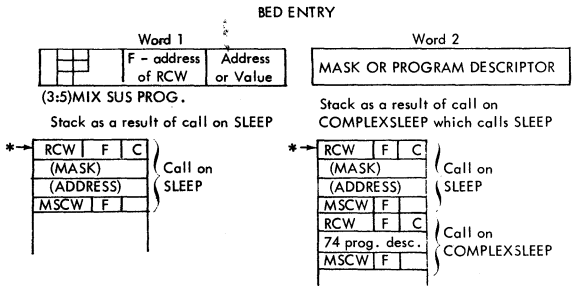
[3:5] M MIX INDEX of suspended program.
[8:10] 0 Size field not used.
[18:15] F F field - Address of RCW of SLEEP procedure.
[33:15] 1 Value to be tested against the result from the procedure called by accessing word 2.

Word 2 7 4 0 0 0 0 F F F F F C C C C C

[0:48] Program descriptor which when accessed will return a value of 1 if the suspended program can be reactivated or 0 (zero) if it cannot be reactivated.

As conditions dictate, NOTHINGTODO searches the BED to determine if a program can be reactivated. Essentially, the following statements indicate how the test is made. (BED is ordered by priority.)

```
NT1←Index of entry to be tested;
NT2←BED [NT1];
NT3←BED [NT1+1];
IF NOT (NT2 AND NT3) ≠ NOT 0 THEN START JOB;
```



DISK COMMUNICATES

The Communicate syllable transfers the "communicate literal" from the top of the stack to R+9 (11 octal) and sets the communicate interrupt bit:

OCT	DEC	
0	0	- Invalid EOJ
1	1	- Time
2	2	- Sleep (wait)
3	3	- Return an array
4	4	- Zip with
5	5	- End of job
6	6	- When
7	7	- Fill
10	8	- Zip
11	9	- Data comm - Fill with inquiry
12	10	- Block Exit (ALGOL Storage Return)
13	11	- ALGOL I/O
14	12	- Break
15	13	- COBOL I/O
16	14	- Data Segment
17	15	- Display
20	16	- Accept
21	17	- COBOL I/O Errors
22	18	- Data Communication Write
23	19	- PBT - Printer Backup Routine
24	20	- COBOL Sort
25	21	- Get Space for Sort
26	22	- Return space in 21
27	23	- Load Control
30	24	- Return one row of disk input file
31	25	- Turn array back and switch with PRT (17) (Interchanges Array desc.)
32	26	- Invalid argument to the LN and SQRT intrinsics
33	27	- COBOL Datacomm Interrogate
34	28	- ALGOL Datacomm Interrogate
35	29	- Used for DS-ing programs (Various errors)
36	30	- Directory Search Statement
37	31	- ALGOL DELAY function
40	32	- Datacomm SEEKs and STATUS

OPTION WORD

Word stored in the MCP PRT specifying the status of options. Options can be set or reset via the COLD START routine or through keyboard input.

```
MOD3IOS      = OPTION. [2:1]
RELTOG       = OPTION. [27:1]
PBDREL       = OPTION. [26:1]
DSKTOG       = OPTION. [28:1]
SECMSG       = OPTION. [29:1]
SCHEDMSG     = OPTION. [30:1]
LIBMSG       = OPTION. [31:1]
RETMSG       = OPTION. [32:1]

CLOSEMESS    = OPTION. [34:1]
COPNMESS     = OPTION. [35:1]
DISCONDC     = OPTION. [36:1]
NOTIFYOP     = OPTION. [36:1]
CLEARWRS     = OPTION. [37:1]
AUTOPRINT    = OPTION. [38:1]
SAMEBREAKTAPE= OPTION. [39:1]
GIVETIME     = OPTION. [40:1]
GIVEDATE     = OPTION. [41:1]
TERMGO       = OPTION. [42:1]
OPNMESS      = OPTION. [43:1]
EOJMESS      = OPTION. [44:1]
BOJMESS      = OPTION. [45:1]
USEDRB       = OPTION. [46:1]
USEDRA       = OPTION. [47:1]
```

B 5500 STATION TABLE FORMAT

0:1 Flag bit = 0
1:1 (Mix-message ready flag -- for MCP use)
2:2 (Index to segment of SPO message currently being printed -- for MCP use)
4:4 (TU index into STATION for next control station, <own TU index if not a control station> -- for MCP use)
8:1 DTCU absent: P = 0, A = 1
9:4 TU address for this word
13:1 DTCU translator bypassed: T = 1, F = 0 (translation ASCII + BCL or BAUDOT - BCL)
14:4 Buffer address for this word
18:4 (Buffer index into STATION for next control station, <own buffer index if not a control station> -- for MCP use)
22:1 Station busy
23:1 Adaptor sensed "abnormal" condition
24:1 Buffer is Read-ready
25:1 Type of ending on input message: GM = 0, Full buffer = 1

PRINTED IN U.S. AMERICA 10-67 1031986

26:1 Break: If TRUE, then break key on typewriter or TWX pressed during output.
27:1 Write ready: If TRUE, then write was performed without GM ending. (Additional write required to clear buffer.) If FALSE then GM ending.
28:1 Input-error
29:1 Write-in-process
30:1 Station not ready
31:1 (Mix-messages requested flag -- for MCP use)
32:1 (Mix-message waiting flag -- for MCP use)
33:5 (Mix index of job for which mix-messages have been requested -- for MCP use)
38:5 (Exclusive user's mix index -- for MCP use)
43:1 (Tanked input -- for MCP use)
44:1 (Tanked MCP input being entered -- for MCP use)
45:1 (Station assigned to a job -- for MCP use)
46:1 (Station logged in -- for MCP use)
47:1 (SPO-type message in process -- for MCP use)

PROGRAM AND DUMP INTERROGATION WORKSHEET
(All mathematics must be in Octal)

Instructions: Fill in the requested questions from the I/O Control panel or from the program dump. These answers will lead you to the MCP or object program area where the trouble occurred and identify that area or program segment for your further analysis.

NOTE: Addr Addressed by or Address of

1. a. When the program hung up, was the Processor operating primarily in the:
CONTROL STATE - NORMAL STATE
(circle one)
b. Is MEMORY CHECK light on console ON?
YES - NO
(circle one)

C = _ _ _ _ _ _
S = _ _ _ _ _ _
M = _ _ _ _ _ _
R = _ _ _ _ _ _
F = _ _ _ _ _ _

2. What is the dominant register addresses?

Note: If not readily visible, use either "Stop on Exit" or "Stop on Interrupt" to obtain an address.

L	H	K	V	G

3. What was the Supervisory Printer print out for this error?

4. ARRAY PRT[*,*] - Cell 235 _____ 5 0 0 0 1 2 0 0 0 0 0 0 | _ _ _ _
5. P1MIX - Cell 226___(P2MIX - Cell 227) _ _ _ _ _ _ _ _ _ _ _ _ | _ _ _ _
6. Address of Descriptor → PRT Base R+0 (add line 4 to 5) | addr-PRT

7. PRT base R+0 must contain EEEEEEEE or 2525252525252525

8. R+10 - INITIATE CONTROL WORD _ _ _ _ _ _ _ _ _ _ _ * | addr-IRCW
(* [32:1] = 1 Chr. Mode; = 0 Word Mode)

9. INTERRUPT RETURN CONTROL WORD _ _ _ | * L | _ _ addr-F _ _ | addr-C
(* [10:2] - The 2 low bits of octade)

10. INTERRUPT CONTROL WORD (IRCW minus 1) _ _ | R-reg | * M S | _ _ _ _ | addr-M
(*[16:1] =MSFF; [17:1] =SALF)

11. INTERRUPT LOOP CONTROL WORD _ _ _ | L | RF | addr-S | addr-C | _ _ _
(ICW-1) (Used for Character Mode only)

JAR * JOBS ACTUALLY RUNNING

12. ARRAY JAR *,* - Cell 236_____ 5 0 0 0 1 2 0 0 0 0 0 0 | _ _ _ _ _
13. P1MIX-Cell 226___(P2MIX-Cell 227___) _ _ _ _ _ _ _ _ _ _ _ _ | _ _ _ _ _
14. Address of Descriptor JAR *,0 the row base (add line 12 & 13) | addr-J ROW

15. JAR *,0 Object programs 1st name (Alpha decode 7 characters)
16. JAR *,1 Object programs 2nd name (Alpha decode 7 characters)

PROGRAM SEGMENT DICTIONARY

17. R+4 Descriptor Segment Dictionary 5 _ _ _ _ _ _ _ _ _ _ | addr-SD

If P1MIX or P2MIX is Not zero (depending on processor) at time of Dump:
Using the "C" register value, search the segment dictionary by an analysis of the "F" field 18:15 of each word for the closest address that is equal to or less than the "C" register value. The Segment number is equal to the relative position of the "found word" in the segment dictionary.

To verify that correct segment has been located:
Obtain Segment size by either;
a. Examine segment base address word minus one (which is 2nd word of in-use link) and obtain prog. seg. size from bits [0-33.]
or
b. Extracting Segment size from Program Listing, ADD segment size to segment base address, checking that address overlaps the "C" register value as copied from display panel.

If P1MIX is zero, the MCP is running.

MCP ROUTINE DETERMINATION

18. ESPBIT Procedure address-Cell 215__ 7 5 6 0 6 2 0 0 0 0 0 0 | addrESPBIT

If "C" register (from display panel) < ESPBIT procedure address THEN routine is "Outer Block Code" ELSE routine is a MCP procedure;

Outer Block Code: Convert "C" register address to Decimal for a direct reference to MCP listing of Outer Block Code.

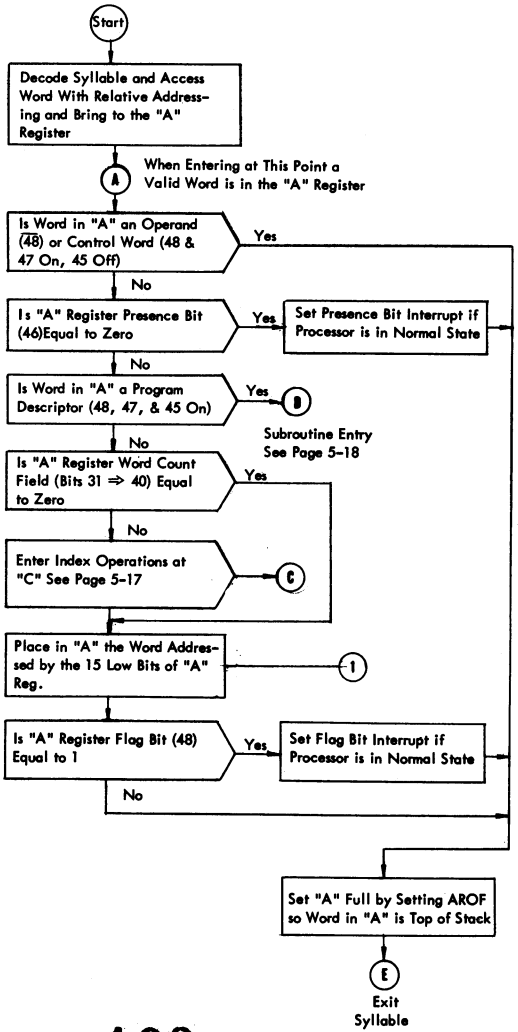
MCP Procedure: Using "C" register value from display panel, search the MCP PRT (starting a cell 00200) until an address plus Word Count encompasses the "C" register value. Reference this address to a "PRT/INDEX" listing to specify the Procedure being executed at time of dump.

The relative position of coding within the procedure may be computed by taking the difference (octal) of the "C" register value and Base address of the Procedure, and converting to decimal; a direct reference to an MCP listing of the procedure is facilitated.

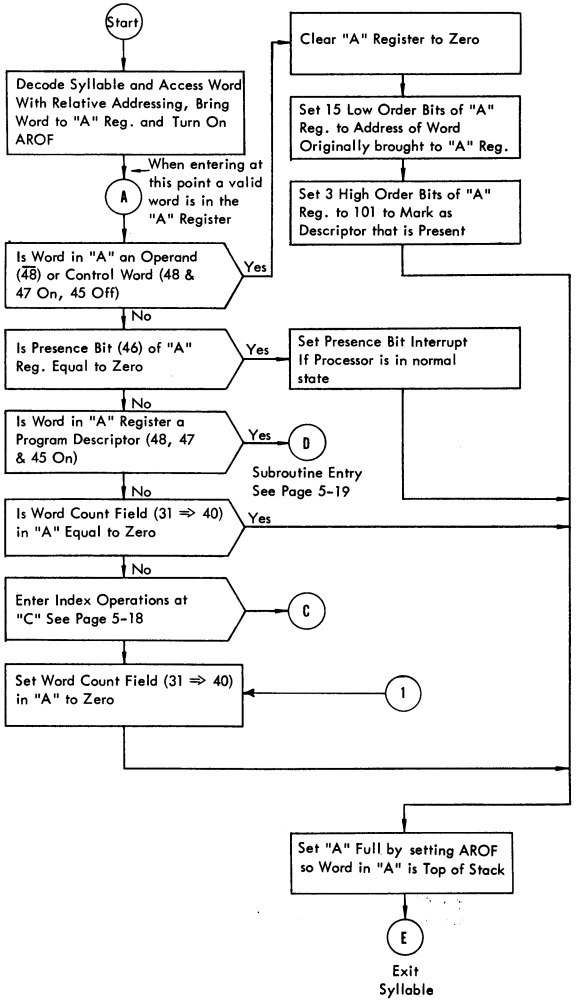
PRINTED IN U.S. AMERICA 10-67

1031986

OPERAND CALL SYLLABLE FLOW CHART



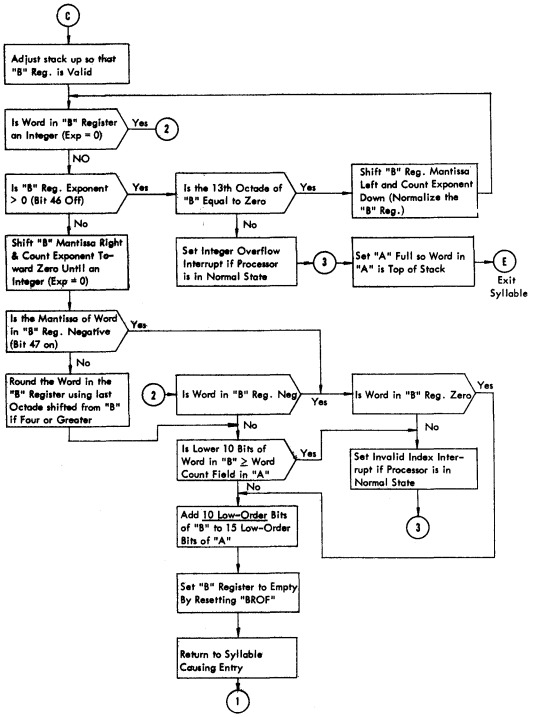
DESCRIPTOR CALL SYLLABLE FLOW CHART



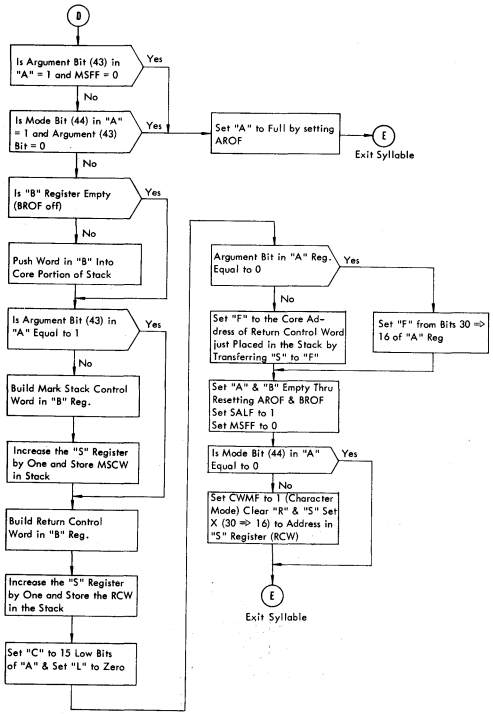
PRINTED IN U.S. AMERICA REVISED 8/70

1031986

INDEX OPERATIONS - OPERAND AND DESCRIPTOR CALL SYLLABLE FLOW CHART

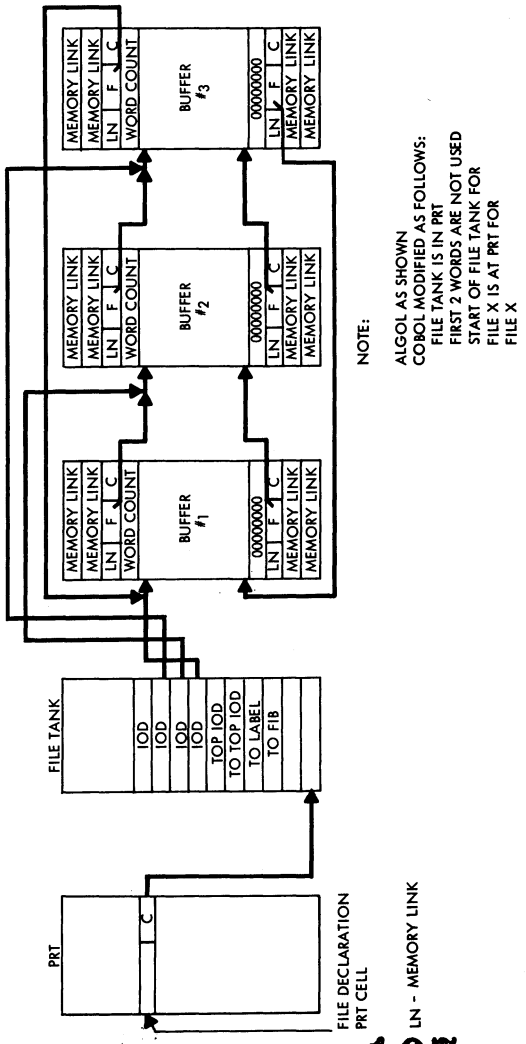


SUBROUTINE ENTRY - OPERAND OR DESCRIPTOR CALL SYLLABLE FLOW CHART



PRINTED IN U.S. AMERICA REVISED 8/70 1031986

FILE BUFFER LOCATION



SECTION 6
MCP GENERAL INFORMATION

MAIN MCP PRT LOCATIONS

The PRT contains the locations reserved for variables, data descriptors, and program descriptors which give information about data arrays and other program information. These locations are likely to change in future MCP's. They are:

	Description
MEMORY	5000 0000 0000 0000
RRRMECH	Mask word used by STATUS to check I/O devices.
[SLATE]	Descriptor pointing to SLATE array.
NSLATE	Pointer to last entry which was started from SLATE.
LSLATE	Pointer to last entry placed in the SLATE.
ESBIT	MCP PRESENCE BIT PROCEDURE
AVAIL	Contains the address of the stopper for available storage links, its value is the highest available address -1.
MSTART	Contains the address of the first area of storage after end of ESP program.
MEND	Pointer to last storage link in memory.
TOGLE	HP2T06, STATUSBIT, SHEETFREE, STACKUSE, STOKEDY, USERSPACEREADY, HOLDFREE, NSECONDREADY, ABORTABLE, BUMPTUTIME, KEYBOARDREADY, NOBACKTALK. QTRDY, INTFREE, SPOEDNULLOG, REMOTELOGFREE.
[BED]	Descriptor pointing to BED array.
P1MIX	Mix index for the job currently being processed. P1MIX = 0 means no job is currently being processed.
P2MIX	Mix index for the job being currently processed on Processor 2. If no Processor 2 then P2MIX = -1.
DATE	Contains current date (YYDDD -- BCL).
CLOCK	Contains (the number of "time interval interrupts" processed since halt-load) multiplied by 64.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

XCLOCK	External clock (clock which is set by system operator and tells time for day).
READY	Contains the contents of the ready register on the last read.
[PRT]	Descriptor pointing to PRT array.
[JAR]	Descriptor pointing to JAR array.
[SHEET]	Descriptor pointing to SHEET array.
[JOBNUM]	Pointer to last entry in BED.
[PRYOR]	Table containing priorities for each mix index.
NOPROCESSTOG	<0 if normal state processing is allowed.
[NFO]	Pointer to NFO array.
[ISTACK]	Independent stack (Stack use true if independent Stack is not in USR).
[PROCTIME]	PROCTIME [I] contains processor time for job with mix index = I.
[IOTIME]	IOTIME [I] contains I-O time for job with mix index = I.
[CHANNEL]	CHANNEL [I] contains logical unit of last descriptor sent out on channel I.
[FINALQUE]	Pointer to FINALQUE.
[LOCATQUE]	Pointer to LOCATQUE.
[IOQUEAVAIL]	Pointer to first available space in IOQUE.
[IOQUE]	Pointer to IOQUE array.
[UNIT]	Pointer to UNIT array.
[TINU]	Pointer to TINU array.
[WAITQUE]	A QUEUE of units for which there are I/O requests but no I/O channel is available.
NEXTWAIT	Pointer into WAITQUE at next available slot.
FIRSTWAIT	Pointer at next unit to be used when a channel becomes available.
[LABELTABLE]	Pointer to LABELTABLE array.
[MULTITABLE]	Pointer to MULTITABLE array.

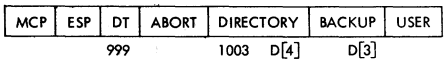
[RDCTABLE]	Pointer to RDCTABLE array.
PRNTABLE	
OPTION	Contains option word.
ILL	Used to link together tanked output for Datacomm.
INQCT	Counter of unprocessed Datacomm interrupts.
PINGO	Used to link together tanked input messages for the MCP.
READQ	Used to link together tanked input for Datacomm.
RRNCOUNT	A counter incremented for each "Read Ready Normal" Datacomm interrupt.
[DMIX]	Pointer to DMIX array. Indexes into DALOC.
[DALOC]	Pointer to DALOC array.
STATUS	STATUS PROCEDURE (STATUSBIT FALSE if status routine is not running).
CORE	[4:14] MULTIPROC FACTOR [18:15] (Sum of CORE est. for all jobs active in MIX) DIV 64 [33:15] (Amount CORE initially avail for all jobs) DIV 64.
KEYBOARDCOUNTER	Counter of unprocessed keyboard requests.
NUMESS	
STATIONMESSAGEHOLDER	
STATION	
FS	Two dimensional array by mix index containing negation of the "Type of User" Code for each file.
TUMAX	Number of Terminal Units.
ATTACHED	Two dimensional array by mix of station address attached to the mix.
USERCODE	Table of in-use "User Codes" by mix index (if <0 the file handling is to be performed as if no user code present) - temporarily (Status 3)
LOOKQ	Used to link together "User Codes", Mask words and Times for remotes.

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

UNITCODE	Table of "User Codes" for input devices (13 words long).
MCP	Privileged M user.
Mixmask	Mask for allowable input mix-messages from remote.
Infomask 1 Infomask 2	Mask for allowable keyboard input messages from remotes not requiring a mix index.
CCmask 1 CCmask 2	Mask for allowable control card reserved words from remote stations.

DISK LAYOUT



MCP - Interrupt code
PRT
Outer Block code
Save code
Non-Save code
} MCP on Disk with relative positions as illustrated.

ESP - "Executive Scratch Pad" used by MCP for Scratch Pad area.

DIRECTORYTOP - (seg. 999) Contains parameters for MCP
- D[0] = OPTION WORD
- D[1] = DATE
- D[2] = NUMBER OF ELECTRONIC UNITS
- D[3] = HIGHEST ADDRESS OF BACKUP STORAGE
- D[4] = HIGHEST ADDRESS OF DIRECTORY
- D[5] = LAST NUMBER USED FOR CONTROL DECK
- D[6] = FIRST CONTROL DECK QUEUED (LOCATION IN DIRECTORY)
- D[7] = LAST CONTROL DECK QUEUED (LOCATION IN DIRECTORY)
- D[8] = NEXT NUMBER AVAILABLE FOR PRINTER BACKUP DISK
- D[9] = CORE, CONTAINS MULTIPROCESSING FACTOR
- D[10] THRU D[15] SPECIFY WHICH DC-STATIONS ARE SPO-LIKE
- D[16] = Q VALUE OF READ-READY LIMIT FOR "DATACOM SPOS".

ABORTABLE - (at DIRECTORYTOP+1) Used by NSECOND in termination to log off ABORT type jobs.

DIRECTORY - Area used by MCP to maintain Directory of entire disk.

BACKUP - Area used by MCP to store information overlayed from memory.

USER - Area used for storage of the system log, compilers, and user files.

ABORTABLE format:

First Segment:

Word	Contents
0	XCLOCK
1	DATE
2	"ABORT"

The next three entries are repeated for each job in the mix. Entries are zeroed if mix number is not assigned.

3	Process Time
4	I/O Time
5	IDLETIME (from the JAR)

Second Segment:

Word	Contents
0-3	Not assigned
4	First name of object program
5	Second name of object program
6	Start time
7	Pointer to location of control card in ESPDISK to be written into the SYSTEM LOG.

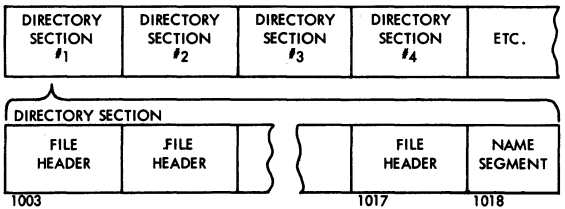
The above four entries are repeated for each mix index assigned.

DISK DIRECTORY

The DF MCP maintains, on disk, a Disk Directory which provides information about all permanent files on the disk. The Directory consists of Sections, where each Section is composed of 16 segments which contain information for up to 15 files. These Sections are allocated as needed in the Disk Directory.

The 16th segment in a section contains the names (i.e., file identification) of each file defined in that section. The remaining 15 segments are referred to as file headers.

DISK DIRECTORY



Name Segment

Contains up to 15 pairs of names for each file defined in this section (i.e., Multifile ID / File ID or Program ID / Program ID Suffix). NOTE: An entry of @14 in the first word of a two word entry position denotes this position available for an entry. An entry of @114 denotes the last entry of the Directory.

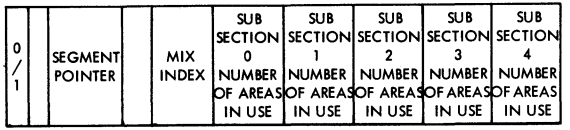
PRINTED IN U.S. AMERICA 10-67 1031986

File Header

Word	
0	[0:15] Record length [15:15] Block length [30:12] Record/Block [42:6] Segments/Block
1	Row length
2	[0:48] = 0 Free File [1:1] = 0 Sole User, Public or Private File [1:1] = 1 Security File [6:42] = primary user's user code
3	[0:18] Save Factor (BCL) [18:30] Creation Date (BCL)
4	[1:1] = 1 if file interlock [12:30] Date of last access (BCL) [42:6] Open Count
5	Sole User File = 0; Public File = "?"; Private File = 1:1 = 1, 6:42 = Multi-file ID of Security File.
6	Sole User File = 0; Public File = 0; Private File = 1:1 = 0, 6:42 = File ID of Security File.
7	END OF FILE COUNT (number of records)
8	ROW LENGTH as specified in file declaration
9	Number of rows as specified in file declaration
10-29	Disk addresses of each row if assigned (binary)

DALOC-ARRAY DALOC[*]SIZE IS 64 OR @ 100

This table is used to keep track of backup disk. An index to DALOC is provided from DMIX. All words in table DALOC have the format noted below. The first table entry is in element 1. Each one-word entry keeps track of a 500 segment which is divided into sub-sections of 100 segments. The element number of the entry is used to calculate the absolute address of the beginning of a section.

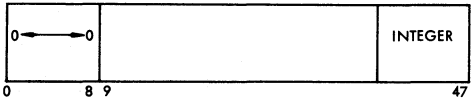


[0:1]	0	Flag bit (0, if area is used; 1, if area is not used).
[2:7]	0-99	Relative address within sub-section of next available segment (pointer advances only).
[9:3]	0-4	Number of sub-section currently being used.

[12:6]	0-63	Mix INDEX of program currently using this section.
[18:6]	0-63	Number of areas, within the sub-section, which are in use (i.e., number of writes in sub-section minus number of reads from sub-section).
24:6]	0-63	
[30:6]	0-63	
[36:6]	0-63	
[42:6]	0-63	

DMIX-ARRAY DMIX[*]SIZE IS MIXMAX+1

DMIX is a table of indexes into the table DALOC. Each program in a current mix has an entry in DMIX in the DMIX element corresponding in number with the program's mix INDEX. (E.g., A program with the MIX INDEX MNDX has an entry in DMIX [MNDX].) All entries in DMIX have the format noted below.



[0:9]	0	
[9:39]	Index to DALOC	(ZERO if no disk has been assigned to the program)

AVAILABLE-DISK TABLE

The Available-Disk Table is a list containing an entry for each area of available disk storage. The list is composed of one or more segments, depending upon the number of available areas. The list is maintained in memory order (i.e., each list entry following the first entry defines an area with greater address).

Word		Contents
0	[18:15]	Number of available areas for this section
	[33:15]	Link to next segment of available disk space table
1-29	[5:18]	Number of segments of disk space
	[23:25]	Address of disk area

SLATE[*]

The SLATE is a queue of requests to run independent DF MCP routines (i.e., routines whose functions are not directly related to object programs; e.g., STATUS, CONTROLCARD, SELECTION and RUN).

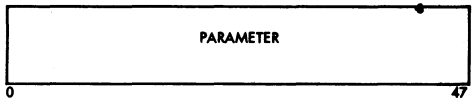
DF MCP routines which desire to run independent routines cause entries to be made in the SLATE by calling the INDEPENDENTRUNNER routine and passing the address of the program descriptor for that routine and a parameter for the routine. INDEPEN-

PRINTED IN U.S. AMERICA 10-67 1031986

DENTRUNNER then makes the two necessary entries into the SLATE. The first word of an entry is a parameter to the routine. The second word of an entry is the PRT address of the routine. NSLATE and LSLATE are pointers into the SLATE. NSLATE points at the last entry which was started, and LSLATE points at the last entry placed in the SLATE.

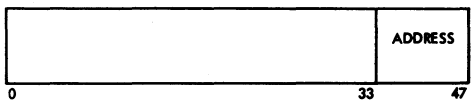
Routines noted in the SLATE are called out by the NOTHINGTODO routine on a first-in, first-run basis. All entries in the SLATE have the format noted below.

WORD 1



[0:48] Varies according to routine. Parameter for Independent routine.

WORD 2



[0:33] 0

[33:15] Address Address points to program descriptor of independent routine.

SHEET

The SHEET provides information to the SELECTION routine to introduce jobs into the MIX. The PRT cell "SHEET" gives the disk address of the first sheet entry. SHEET is the storage area that is used to store program parameters prior to the program being placed in the MIX at which time portions of the SHEET are placed in JAR.

ENTRIES IN THE SHEET ARE AS FOLLOWS:

S[0] = 1ST NAME (7 CHRS)
S[1] = 2ND NAME (7 CHRS)
S[2] . [1:2] = 0 NORMAL
= 2 JOB HAS BEEN XS-ED (FORCED RUN)
= 3 JOB HAS BEEN ES-ED (FORCED RUN AND DS)
S[2] . [8:10] = 0 GO JOB (FROM COMPILE & GO)
= 1 COMPILER (FOR COMPILE & GO)
= 2 EXECUTE JOB
= 3 COMPILER (FOR SYNTAX CHECK) (SET TO 2 LATER)
= 4 COMPILER (FOR COMPILE TO LIBRARY)
= 5 RUN JOB
S[2] . [8:15] = SKELETONS DISK ADDRESS (IF S[2] . [8:10] = 1,2,4)
S[2] . [33:15] = PRIORITY, SAME AS S[18]
S[3] . [8:10] = SCHEDULE-ID FOR THIS JOB
S[5] = STARTING TIME FOR LOG
S[6] = LOCATION OF LAST PART OF LOG

S[13] = DISK ADDRESS OF LABEL EQUATION ENTRIES APPLICABLE TO THIS EXECUTION ONLY
S[15] = DISK ADDRESS OF LABEL EQUATION ENTRIES PRESENTED WHEN PROGRAM WAS COMPILED AND APPLICABLE TO ALL EXECUTIONS (SEE BELOW)
S[16] = ESTIMATED PROCESSOR TIME
S[17] = ESTIMATED I/O TIME
S[18] = PRIORITY
S[19] = COMMON VALUE
S[20] = ESTIMATED CORE REQUIREMENTS
S[21] = STACK SIZE
S[22] = TIME TO SAVE PROGRAM ON COMPILE TO LIBRARY
S[23] . [9:9] = REMOTE STATION ADDRESS, ELSE 0
S[23] . [31:17] = TIME JOB WAS PUT INTO SHEET (FOR TS MESSAGE)
S[24] = USER CODE
S[29] = DISK ADDRESS OF NEXT SHEET ENTRY (=0 IF LAST)

ENTRIES FOR LABEL EQAT ARE AS FOLLOWS:

F[0] = MULTI-FILE ID (7 CHRS)
F[1] = FILE ID (7 CHRS)
F[2] . [0:18] = REEL NO (3 CHRS)
F[2] . [18:30] = CREATION DATE (5 CHRS "YYDDD")
F[3] . [0:12] = CYCLE (2 CHRS)
F[3] . [41:1] = OPERATOR NOTIFICATION BIT
F[3] . [42:6] = 0 FOR CP (FILE TYPES)
1 FOR LP
2 FOR MT
3 FOR SPECIFIC UNIT
4 FOR LP (MAY BACKUP)
5 FOR SPECIFIC (UNLABLED)
6 FOR LP (MUST BACKUP)
7 FOR PT
8 FOR PT (UNLABLED)
9 FOR MT (UNLABLED)
10 FOR DISK
11 FOR SPO
12 FOR DISK SERIAL
13 FOR DISK UPDATE
14 FOR DATA COMMUNICATION
F[4] . [0:6] = NO OF CHARS IN INTERNAL NAME
F[4] . [6:42] = INTERNAL NAME (MAY CONTINUE TO F[11])
F[14] -[F 25] SAME AS ABOVE FOR NEXT FILE (F[14] = 14 IF NO NEXT)
F[29] = DISK ADDRESS OF NEXT LABEL EQAT. ENTRY (= 0 IF NONE)

CONTENTS OF JAR ARE:

J[0] - J[6] = SAME AS SHEET ENTRIES S[0] - S[6]
J[7] = IDLE TIME
J[8] = LENGTH OF EACH ROW OF CODE FILE
J[9] = NO. OF ROWS
J[10] - J[29] = DISK ADDRESS OF ROWS

PRINTED IN U.S. AMERICA REVISED 8/70

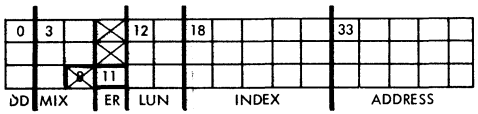
1031986

I/O-QUEUE

IOQUE, FINALQUE, and LOCATQUE together with UNIT forms the I/O-QUEUE. An I/O request for logical unit U requires three words of space in the I/O-QUEUE. If the request occupies position S in the I/O-QUEUE, then IOQUE[S] contains the I/O descriptor for the request. FINALQUE[S] contains the I/O descriptor skeleton to be used at I/O complete time to rebuild the original I/O descriptor. LOCATQUE S points to the location of the I/O descriptor at the time of request. The spaces not used in the I/O-QUEUE are linked together through IOQUE. The first available entry is pointed to by IOQUEAVAIL.

ARRAY LOCATQUE [*] ;

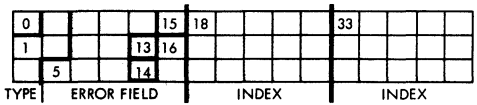
All entries in LOCATQUE have the following format:



[0:3]	5	Descriptor identification bits.
[3:5]	MIX INDEX	MIX INDEX of program which requested the I/O operation.
[8:1]	SHARED DISK	
[9:1]	DISABLE ERROR MESSAGE DISK	
[10:1]	TSSMCP	
[11:1]	ER	Error recovery in process on this I/O.
[12:6]	LUN	Logical unit number of unit on which the I/O is to be executed.
[18:15]	INDEX	Index into I/O-QUEUE of next I/O request on this unit. @77777 if no additional requests.
[33:15]	ADDRESS	Address of the I/O descriptor used for this request, at time request was made. If buffering is being used by the object program, the descriptors are rotated and the I/O descriptor may not remain in its original location.

ARRAY UNIT [32] ;

All entries in UNIT have the following format:



[0:1]	0	Flag bit.
[1:4]	Type Code	0 = Card Reader; 1 = Line Printer; 2 = Magnetic Tape; 3 = Drum; 4 = Disk; 5 = SPO; 6 = Card Punch; 8 = Paper Tape Punch; 9 = Paper Tape Reader; 10 = DATA COMM.
[5:8]	Error Field	Copy of last error field of last I/O RESULT DESCRIPTOR on this unit.
[9:4]	Temporary Error Count	

[13:1]	0/1	Not ready bit 0 = unit ready; 1 = unit not ready
[14:1]	0/1	Error flag bit 0 = no errors; 1 = errors
[15:1]	0/1	Waiting for I/O channel 1 = I/O awaiting an I/O channel.
[16:2]	varies	I/O in Process bits 00 = unit not in process; 11 = unit in process; 01 = (for line printer only) I/O complete but awaiting printer finish.
[18:15]	INDEX	Index of first I/O request for which service is not complete. @77777 if none.
[33:15]	INDEX	Index of last I/O request for which service is not complete.

INPUT OUTPUT ASSIGNMENT TABLES

LOGICAL UNIT NUMBERS

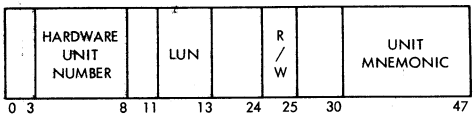
The DF MCP associates one unique logical unit number with each I/O unit (this is different from the hardware unit numbers). The logical unit numbers assigned the I/O units were determined by the format of the result of the Read Ready Register (RRR) operator. The result of the RRR operation is stored in the fields [17:31]. Numbering from right to left, bit [47:1] is numbered 0, and bit [22:1] is numbered 25.

UNIT	LOGICAL UNIT NUMBER	UNIT	LOGICAL UNIT NUMBER	UNIT	LOGICAL UNIT NUMBER
MTA	0	MTK	8	DRA	16
MTB	1	MTL	9	DRB	17
MTC	2	MTM	10	DKA	18
MTD	3	MTN	11	DKB	19
MTE	4	MTP	12	LPA	20
MTF	5	MTR	13	LPB	21
MTH	6	MTS	14	CPA	22
MTJ	7	MTT	15	CRA	23
				CRB	24
				SPO	25
				PPA	26
				PRA	27
				PPB	28
				PPB	29
				DCA	30
				MCP	31
			PSEUDO	RDRS	32+

TINU

TINU is an array used by the I/O routines and provides information about the I/O units. The entries in TINU are ordered according to logical unit number. That is, information about the unit with logical unit number, LUN, would be in TINU[LUN]. All entries in TINU have the following format:

PRINTED IN U.S. AMERICA REVISED 8/70 1031986



[0:3]	0	Not used.
[3:5]		Unit number recognized by hardware.
[8:5]	0	Not used.
[11:7]		This field contains the logical unit number indicator, which has the following characteristics. The expression (0&TINU[LUN] . [5:11:7]/@1000000000000) will produce a result with all zeroes except in the bit location corresponding to the RRR result bit location designated for the unit represented by TINU[LUN].
[18:6]	0	Not used.
[24:1]	0/1	This bit indicates the setting of the R/W bit in the initial I/O descriptor for the unit. 0 = in, 1 = out. This setting would be used for RELEASE statements.
[25:5]	0	Not used.
[30:18]	Unit mnemonic	Three character abbreviation for the unit represented by TINU[LUN]. (E.g., CRA.)

LABELTABLE, MULTITABLE, and RDCTABLE contain label information by logical unit number.

LABELTABLE[I] contains the file-id. for logical unit I.
MULTITABLE[I] contains the corresponding multi-file id.
RDCTABLE[I] contains the corresponding reel number ([14:10]), reaction date ([24:17]), and CYCLE ([41:7]). If UNIT I is assigned to a program, RDCTABLE[I], [8:6] contains mix index. Special entries into the LABELTABLE include:

@114	Unit not ready
-@14	Unit in use by CONTROLCARD
@214	Unit is RW/LK
@314	Unit contains an unlabeled tape
+	Unit available
-	Unit in use
0	Scratch

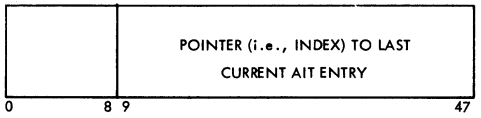
For units 0 through 15:

PRNTABLE[I] contains; If assigned to a program, the address of the top I/O descriptor in [15:15]. PRNTABLE[I] . [1:1] is 1 if the unit has a write ring.

ARRAY INFORMATION TABLE

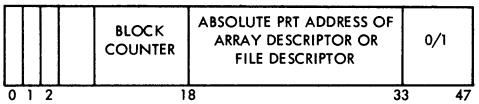
One AIT is associated with each ALGOL program that declares one or more files or arrays.

WORD 0



[0:9]	0	
[19:39]	INTEGER	Index to last current AIT entry.

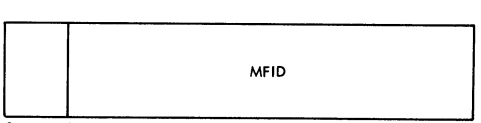
REMAINING WORDS



[0:1]	0	FLAG BIT
[1:2]	0,1,2	(0 = ARRAY, 1 = RUN-TIME ERROR ENTRY 2 = FILE)
[3:5]		Number of dimensions
[8:10]	0-1023	Block counter (i.e., nesting depth when file or array is declared)
[18:15]	Address	Absolute address of file or array descriptor or of RTE (RUN TIME ERRORS) cell containing label
[33:15]	0/1	
	1,2,4,8,16	Save indicator (1=SAVE) for arrays - for RTE, error type: 1- INTEGER OVERFLOW 2- EXPONENT OVERFLOW 4- INVALID INDEX 8- DIVIDE BY ZERO 16- FLAG BIT

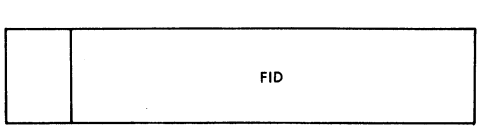
FILE PARAMETER BLOCK (FPB) (ADDRESSED BY R+3)

WORD 1



[0:6]	0	Not used
[6:42]	MFID	Seven characters multi-file identification

WORD 2

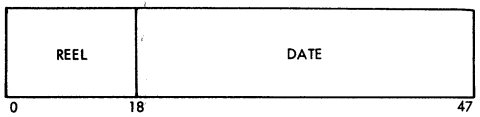


[0:6]	0	Not used
[6:42]	FID	Seven character file identification

PRINTED IN U.S. AMERICA REVISED 8/70-

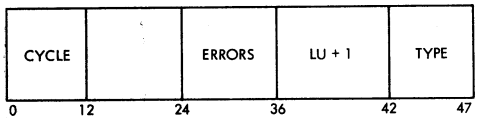
1031986

WORD 3



[0:18]	REEL	Reel number in three character alpha
[18:30]	DATE	Creation date in five characters

WORD 4



[0:12]	Cycle	Cycle number (two characters)	
[24:12]	Error	Number of errors	
[36:6]	LU+1	Logical unit number plus one	
[42:1]	Forms	Zero indicates unit not assigned	
[42:5]	TYPE	0 = CP/CR	10 = Disk
		1 = LP	11 = SPO
		2 = MT	12 = Disk Serial
		3 = DG - DESIGNATED	13 = Disk Update
		4 = LP/PBT	14 = Date Communication
		5 = Specified unit (unlabeled)	15 = PBD only
		6 = PBT only	16 = PBT/PBD
		7 = PT	17 = LP/PBD
		8 = PT unlabeled	18 = LP/PBT/PBD
		9 = MT unlabeled	

WORD 5

[1:1] FILE OPEN
I/O TIME/UNIT

FILE TANK

ALGOL (Addressed by a descriptor located in the file's PRT cell.)

Word	Contents
0	Not used
1	Not used
2	Pointer to FIB 0
3	Pointer to read-in label if input Pointer to compiler label if output
4	Pointer to top I/O descriptor
5	Top I/O descriptor
6	Remaining I/O descriptors
⋮	
N	

COBOL

Words 2 through N are located in the PRT for COBOL object programs.

FILE INFORMATION BLOCK (FIB)

Word		Contents
0		Beginning file
1		Beginning reel
2		Ending file
3		Ending reel
4	[1:1]	1 = USE routines present
	[2:1]	1 = labels omitted
	[3:2]	EOR rerun: 00 = No, 01 = output tape, 10 = scratch
	[5:1]	1 = optional
	[6:1]	1 = No IN/OUT part
	[7:1]	1 = sort file
	[8:4]	Internal type code
	[12:1]	0 = bits 13:11 is file number 1 = bits 13:11 is FPB index
	[13:11]	See above
	[24:1]	1 = release unit at CLOSE
	[25:2]	Disposition of file 00 = rewind 01 = no rewind 10 = RW/LK 11 = RW and release
	[27:3]	Access mode 0 = serial, 1 = random, 2 = update
	[30:18]	Save factor
5	[40:1]	1 = at end of file
	[1:1]	Indicates an INV USER as opposed to a parity.
	[41:1]	1 = CLOSED, unit retained
	[42:1]	1 = CLOSED, unit released
	[43:1]	1 = input
	[44:1]	1 = reverse
	[45:1]	Not used
	[46:2]	0 = unblocked 1 = TECH A 2 = TECH B 3 = TECH C
6		Block count
7		Record count
8	[3:15]	Relative PRT location of descriptor for hash totals
	[18:15]	Number of rows } DISK FILES
	[33:15]	Size of rows } DISK FILES
9		Rerun control (number of records)
10		Rerun control counter
11		Number of records per block
12		Number of records in current block
13	[1:9]	Number of buffers requested
	[10:9]	Number of buffers assigned
	[19:1]	1 = bad key
	[20:1]	1 = seek given
	[21:1]	1 = read (1st operation)

PRINTED IN U.S. AMERICA 10-67 1031986

Word		Contents
	[22:1]	1 = open
	[23:1]	1 = write block back
	[24:1]	0 = alpha (mode)
	[25:1]	1 = reverse (direction)
	[26:1]	1 = memory inhibit (for input)
	[27:1]	1 = input
	[28:10]	Current reel number
	[38:1]	1 = forms
	[39:5]	External type code
	[44:3]	Not used
	[47:1]	1 = COBOL
14		Descriptor for disk file header in core - 30 words if file open
15		Error use input index
		Error use input end index
	[24:6]	Logical unit number
	[30:10]	Special select counter
	[40:8]	Block count
*16		Copy of current original I/O descriptor
17		Number of words left in the buffer
18	[3:15]	Buffer size
	[18:15]	TECH C buffer length
	[33:15]	Maximum record length
*19		Final I/O descriptor for program release (FINALQUE)

Internal type codes (used in FIB 4 . 8:4)

0 = CR	7 = PBT
1 = LP	8 = PP
2 = MT	9 = PR
3 = DR	10 = DC
4 = DK	11 = CD
5 = SPO	12 = PBD
6 = CP	

NFO

NFO contains the following for each active mix index and is used for reconstructing the PRT for stack overflow conditions. NDX represents the number of entries per job in the NFO table.

NFO [(MIX-1) times NDX] = FILE PARAMETER BLOCK data descriptor R+3
NFO [(MIX-1) times NDX+1] = SEGMENT DICTIONARY name descriptor R+4
NFO [(MIX-1) times NDX+2] = Location of bottom of stack (word containing all B's)

* WITH FLAG BIT OFF.

STANDARD B 5500 LABLE RECORD

Word	Character (word)	Character (record)	Field Description
1	1-8	1-8	Must contain bLABELbb.
2	1	9	Must be zero.
2	2-8	10-16	Multi-file id.
3	1	17	Must be zero.
3	2-8	18-24	File id.
4	1-3	25-27	Reel-Number (within file).
4	4-8	28-32	Date-Written (creation date).
5	1-2	33-34	Cycle-Number (to distinguish between identical runs on the same day).
5	3-7	35-39	Purge-Date (date this file can be destroyed).
5	8	40	Sentinel (1 = End-of-Reel, 0 = End-of-File).
6	1-5	41-45	Block-Count.
6-7	6-8/1-4	46-52	Record Count.
7	5	53	Memory-Dump-Key (1 = memory dump follows label).
7-8	6-8/1-2	54-58	Physical Tape Number.

The remainder of the information contained in the label record varies for ALGOL and COBOL files as follows:

ALGOL FILES

Word	Character (word)	Character (record)	Field Description
8	3	59	Blocking Indicator (3 = blocked, 0 = not blocked)
8	4-8	60-64	Buffer Size (number of words).
9	1-5	65-69	Maximum Record Size (number of words).
9	6-8	70-72	Zeroes.

COBOL FILES

Word	Character (word)	Character (record)	Field Description
8	3-8	59-64	Reserved for File-Control-Routine -- not currently being used.
9-?	1-?	65-??	Users Portion (may be of any format desired by user and may be up to 8,120 characters in length for tape files, up to 16 characters in length for card file, and up to 56 characters in length for printer files.

LOG MAINTENANCE

Log information for programs run on a B 5500 System is written in a file on user disk. The log file occupies one area on disk, and has the <file identification prefix> SYSTEM and the <file identification> LOG. It is the user's responsibility to provide this file.

PRINTED IN U.S. AMERICA 10-67 1031986

The file SYSTEM/LOG is blocked. There are six logical records per physical record. The logical records are five words (i.e., 40 characters) in length; the physical records are 30 words in length.

LOG ENTRY SPECIFICATIONS

Entries in the log can be considered to fall into one of three categories:

a. Compile and go entries.
b. Compile only entries.
c. Execute entries.

With respect to these categories, the following rules determine how a program will be entered in the log:

a. If a compile-and-go is made and the program being compiled contains no syntax errors, the log information for both the compiler and the object program will be listed in a compile-and-go entry.

b. If a compile-and-go run is made and the program being compiled contains syntax errors, if a compile-for-syntax run is made, or if a compile-to-library run is made, the log information for the compiler will be listed in a compile-only entry.

c. If an execute run (i.e., library call out) is made, the log information for the object program will be listed in an execute entry.

The general format of each of the three types of log entries is shown in Figure 6-1. The first log entry starts in the record with relative address 1.

CODE WORD

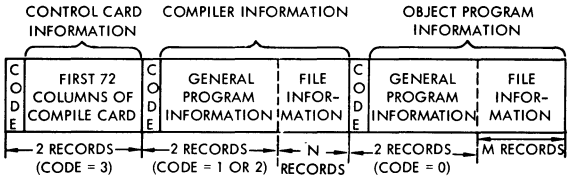
As shown, each log entry contains (1) control card information and (2) compiler and/or object program information. The code word preceding each group of information denotes the type of information. That is, information preceded by a 1 pertains to the ALGOL Compiler; information preceded by a 2 pertains to the COBOL Compiler; and, information preceded by a 3 pertains to an object program. Code 4 denotes the end of log information, while code 5 pertains to printer backup information.

CONTROL CARD INFORMATION

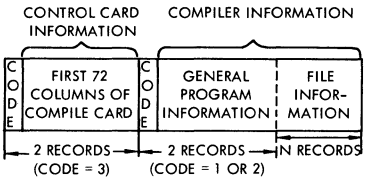
Control card information is contained in the first two records of a log entry, starting at the second word of the first record. This information is a copy of the contents of the first 72 columns of the COMPILE card or EXECUTE card that caused the particular run to be scheduled.

The word immediately preceding control card information is a code with the integer value 3.

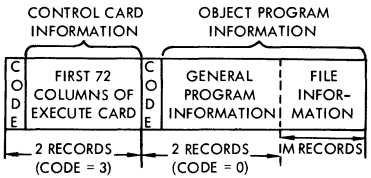
COMPILE AND GO ENTRY



COMPILE ONLY ENTRY



EXECUTE ENTRY



NOTE

N = Number of files declared by compiler.
M = Number of files declared by object program.

FIGURE 6-1 Log Entry Formats

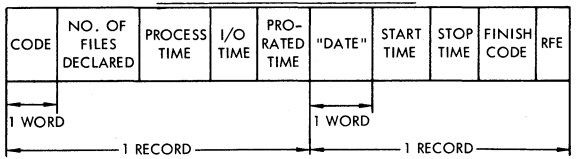
COMPILER AND OBJECT PROGRAM INFORMATION

Compiler information and object program information have identical formats; therefore, the format of this information will be discussed under the general name, program information.

Program information falls into two categories: (1) general program information and (2) file information. The general program information is contained in two records. The file information, which is a copy of the FPB (File Parameter Block) for the program, requires a variable number of records, depending on the number and use of files declared by the program. A record is put in the log for each file declared by the program. If a file is closed more than once or uses more than one physical reel of tape, an additional record appears in the log for each additional closing or tape reel.

The format of a general program information in a log entry (including the code word) is shown in Figure 6-2.

GENERAL PROGRAM INFORMATION

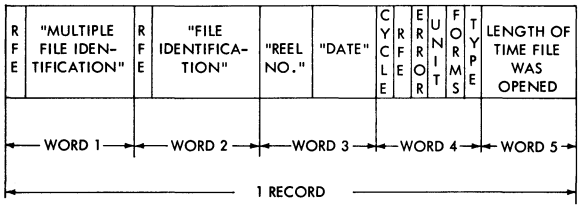


Entry	Description
CODE	INTEGER - 1 = ALGOL, 2 = COBOL, 3 = object program, 5 = printer backup
NO. OF FILES DECLARED	INTEGER
PROCESS TIME	INTEGER - time in 60ths of a second
I/O TIME	INTEGER - time in 60ths of a second
PRORATED TIME	INTEGER - time in 60ths of a second
DATE	BCL - YYDDD format* (e.g., 65046)
START TIME	INTEGER - time in 60ths of a second since HALT-LOAD time
STOP TIME	INTEGER - time in 60ths of a second since HALT-LOAD time
FINISH TIME	INTEGER - 0 = EOJ, 1 = SYNTAX ERROR, 2 = DS-ED, 3 = ABORT
RFE	Reserved for expansion

FIGURE 6-2 FORMAT OF GENERAL PROGRAM INFORMATION

* The YYDDD format provides that the YY characters specify the last two digits of the year, and the DDD characters specify the number of the day of the year.

Figure 6-3 shows the format of one file-information record.



Word	Entry	Field	Contents
1	MULTIPLE FILE IDENTIFICATION	[6:42]	Seven character multi-file identification
2	FILE IDENTIFICATION	[6:42]	Seven character file identification
3	REEL NO.	[0:18]	Reel number in three character alpha
	DATE	[18:30]	Creation date in five characters (YYDDD)
4	CYCLE	[0:12]	Cycle number (two characters)
	ERROR	[24:12]	Number of errors in handling this file (binary)
	UNIT	[36:6]	I/O unit used by this file (binary). This number corresponds to Logical Unit Number plus one.

0 - NOT OPENED
1 - MTA
2 - MTB
3 - MTC
4 - MTD
5 - MTE
6 - MTF
7 - MTH
8 - MTJ
9 - MTK
10 - MTL
11 - MTM
12 - MTN
13 - MTP
14 - MTR
15 - MTS
16 - MTT
17 - DRA
18 - DRB
19 - DKA
20 - DKB
21 - LPA
22 - LPB
23 - CPA
24 - CRA
25 - CRB
26 - SPO
27 - PPA
28 - PRA
29 - PPB
30 - PRB
31 - DCA

	FORMS	[42:1]	1 indicates that special forms were required
(4)	TYPE	[43:5]	Type of file (binary) 0 - CP/CR 1 - LP only 2 - MT 3 - DG (designated) 4 - LP/PBT 5 - specified unit (unlabeled) 6 - PBT only 7 - PT 8 - PT unlabeled 9 - MT unlabeled

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

			10 - disk random 11 - SPO 12 - disk serial 13 - disk update 14 - data communications 15 - PBD only 16 - PBT/PBD 17 - LP/PBD 18 - LP/PBT/PBD 19 - REMOTE
5	FILE OPEN	[1:1]	1 indicates the file is open
	LENGTH OF TIME FILE WAS OPENED	[2:46]	I/O time on this unit in 60ths of a second (binary). Cumulative time from INITIATE I/O to I/O FINISH

SPECIAL RECORDS AND LOG INITIATION

Record Zero

The first record in SYSTEM/LOG (i.e., the record with relative address 0) is used by the MCP when making log entries. The value of the first word in record zero specifies the number of records written in the log. The value of the second word specifies the record capacity of the log. The third and fourth words are used in conjunction with the warning messages supplied by the MCP which signify when the log is half-full and full. The fifth word contains, in BCL, DISKLOG.

Record n + 1

The first word of the record immediately following the last log entry contains a code with the value 4. This record denotes the end of log information, and it is not included in the value contained in the first word record of record zero.

Initiating the Log

If a user program wishes to initiate the log (i.e., set up the log so that the MCP considers the log empty), the following action must be performed:

a. The 1st, 3rd, and 4th words in record zero must be set to zero.
b. The 1st word in record 1 must be set to 4.

REMOTE LOG SPECIFICATIONS

The remote log information for the data communications facilities is written in a file on the user disk. The file has the <file identification prefix> "REMOTE" and the <file identification> "LOG". The file REMOTE/LOG is blocked and must be confined to one area on the disk. There are five logical records per physical record. A logical record is five words in length or forty characters; a physical record is thirty words in length. It is the user's responsibility to provide this file. Logging for data communications is bypassed if the system does not provide a REMOTE/LOG file.

LOG ENTRY SPECIFICATIONS

Entries in the Remote Log can be considered to be of five types:

Type 1 Log-Out Entry
Type 2 Log-In Entry
Type 3 Control Card Entry of less than 32 characters
Type 4 Control Card Entry of 32 characters or more, not greater than 72 characters

Type 5 Job Statistics Entry

Type 1, Type 2, and Type 3 entries each require one logical record in the log. Types 4 and 5 require two logical records per entry.

TYPE 1 LOG-OUT ENTRY

The following information is entered into the file REMOT/LOG when a data communications station logs out:

1 Record	Word 0	[9:9] [42:6]	Station Number ([9:4] = TU, [14:4] = BUF) Code = 1
	Word 1		User Identification (as specified by the FILE SECURITY SYSTEM)
	Word 2		Current Date (YYDDD-BCL)
	Word 3		Time of day at Log-Out
	Word 4		Unused

TYPE 2 LOG-IN ENTRY

The MCP enters the following information in the file REMOTE/LOG when a data communications station logs in:

1 Record	Word 0	[9:9] [42:6]	Station Number ([9:4] = TU, [14:4] = BUF) Code = 2
	Word 1		User Identification (as specified by the FILE SECURITY SYSTEM)
	Word 2		Current Date (YYDDD-BCL)
	Word 3		Time of day at Log-In
	Word 4		Unused

TYPE 3 CONTROL CARD ENTRY (31 characters or less)

The MCP enters the following information -- or Type 4 information -- in the file REMOTE/LOG when a job is selected to run. Every RUN or EXECUTE from a remote station is logged:

1 Record	Word 0	[9:9] [18:24] [42:6]	Station Number ([9:4] = TU, [14:4] = BUF) RUN NUMBER* Code = 3
	Word 1 thru Word 4		Contents of Control Card

* Entries in the file REMOTE/LOG corresponding to entries in the file SYSTEM/LOG have the same RUN NUMBER, where a job's RUN NUMBER is defined to be its start time -- in 60ths of a second -- as specified in the System Log.

PRINTED IN U.S. AMERICA 10-67 1031986

TYPE 4 CONTROL CARD ENTRY (32 characters up to 72 characters)

The MCP enters the following information -- or Type 3 information -- in the file REMOTE/LOG when a job is selected to run. Every RUN or EXECUTE from a remote station is logged:

2 Records	Word 0	[9:9]	Station Number ([9:4] = TU, [14:4] = BUF)
		[18:24]	RUN NUMBER*
		[42:6]	Code = 4
	Word 1 thru Word 9		Contents of Control Card

TYPE 5 JOB STATISTICS

The MCP enters the following information in the file REMOTE/LOG when a station detaches from a job:

Word 0	[2:1]	1 if this station attached by entering an EXECUTE or RUN card; 0 if attached by READ SEEK or WRITE
	[9:9]	Station Number
	[18:24]	RUN NUMBER (as specified in the Type 3 or Type 4 Entry)
	[42:6]	Code = 5
Word 1		User Code
Word 2		First name of the object program (7 characters)
Word 3		Second name of the object program (7 characters)
Word 4		Processor Time in 60ths of a second (i.e., processor time used for this station, out of total used by job)
Word 5		Pro-Rated Time in 60ths of a second (i.e., pro-rated time used by this station, out of total used by job)
Word 6		I/O Time in 60ths of a second (i.e., I/O time used by this station, out of total used by job)
Word 7	[3:21]	Start Date -- Date when job attached to this station (in binary)
	[27:21]	Stop Date -- Date when job detached from station (in binary)
Word 8		Attach Time -- Time when job attached to station
Word 9		Detach Time -- Time when job detached from station

CREATION OF REMOTE LOG ENTRIES

As indicated above, log-in, log-out, and control card entries are made at the time at which they occur. This is possible since the information contained in those entries is immediately available.

* Entries in the file REMOTE/LOG corresponding to entries in the file SYSTEM/LOG have the same RUN NUMBER, where a job's RUN NUMBER is defined to be its start time -- in 60ths of a second -- as specified in the System Log.

The information contained within a Job Statistics entry is accumulated during the time which a remote terminal is attached to a program. The entry is recorded in the Remote Log at the time a program and remote terminal become detached from one another.

The responsibility of dictating which remote station is to be charged for any particular "slice" of a program's processor, I/O, and pro-rated time is strictly that of the object program. The task involved in specifying the station to be charged is, however, an easy one. The procedure involved in slicing times is as follows.

The MCP maintains a table, called USERSTA, which contains one location for each program in the mix. The contents of a given program's location in this table is the station address of the remote station presently specified to be charged for the time used by that program.

When a program enters the mix, its location in the USERSTA table is set to the address of station 0/0, a non-existent remote terminal. (The times assigned to station 0/0 are those which the program does not assign to any given station, i.e., they are unassigned times.) Then from that time until the address in that program's USERSTA location changes, station 0/0 is charged for all processor, I/O, and pro-rated times charged to the program. When the address in the program's USERSTA location changes, the remote terminal whose address is then specified begins being charged for the times assigned to the program, etc.

The way in which a program designates the address to be placed in USERSTA -- i.e., the way in which a program designates the station to be charged -- is to perform either a passive or active interrogate statement referencing the station. (In ALGOL this involves a statement of the form STATUS (TUBUF,0) or STATUS (TUBUF,1); in COBOL it involves a statement such as MOVE FILENAME FROM TU,BUF TO STATUSWORD or MOVE FILENAME FROM TU, BUF AFTER CHECK TO STATUS-WORD.) Each time such an interrogate is performed, the MCP checks to see if the terminal buffer address currently in the program's USERSTA location is different from the one specified in the interrogate statement. If it is, the "old station" is charged with all times since the previous change in USERSTA and the new station is established as the new recipient of time.

It should be noted that if a program wishes to designate certain times as being "unassigned" (i.e., assigned to station 0/0) it should perform a passive interrogate on station 0/0.

Whenever a station is "detached" from a program, a job statistic's entry is recorded in the log. That entry, of course, contains all the times which were allotted to the station in the manner described above.

FILE MAINTENANCE PROCEDURES

To retain information for the file REMOTE/LOG, a FILE CARD group should appear in the COLD START DECK.

The first record of the file REMOTE/LOG (i.e., the record with relative address 0) describes the remainder of the file. Contents of record 0 are:

PRINTED IN U.S. AMERICA 10-67 1031986

Record 0 File REMOTE/LOG	Word 0	Value of word equals the number of logical records written in the file REMOTE/LOG.
	Word 1	Value of word equals the record capacity (in logical records) of the file REMOTE/LOG.
	Word 2 thru Word 4	Reserved for system use

A user program must initiate word 0 of the file REMOTE/LOG to 0 and word 1 to the record capacity of the file. For example, if the FILE card in the FILE CARD group of the COLD START deck has the form FILE REMOTE/LOG,1x1000 then a user program must initiate Record 0, Word 0 to 0 and Record 0, Word 1 to 6000.

The B 5500 operator is notified when the log is half-full and when the log is full. Should the log become full, wrap-around will occur. If the log is not present, the operator will be notified the first time the log is accessed.

Operator notification is via the SPO and the messages are:

#REMOTE/LOG FULL
This message will be typed when the log is full. Wrap-around will occur the next time the log is accessed.

#DUMP REMOTE/LOG
This message will be typed when the log is half-full.

#NULL REMOTE/LOG
This message will be typed the first time the remote log is accessed and not present.

PRINTER BACKUP INFORMATION

FORMAT OF BLOCKS IN A PB FILE

Each block of a PB file (any printer-backup file) is 90 words in length, containing five records (except that the last block of a print file -- the logical file as declared and created by the object program -- may contain up to four "garbage" records, which will be ignored). The records are packed into the block in inverted sequence; the first record is in words 72-89, the second in 54-71, the fifth in 0-17.

FORMAT OF RECORDS ON PB FILES

There are two types of records on a PB file: control records and data records. All records are 18 words.

There is one control record per print file, containing in order the file identification, the name of the program creating the print file, a copy of the first nine words of the header card, and a "special forms" flag. This record is the first record of the print file and is the first record of a block on the PB file. A control record is always the first record of a block.

The remainder of the print file is composed of data records; each data record contains a print record created by the program followed by a control word. The control word is a copy of a print descriptor, with the following changes:

a. The continuity bit is OFF, except for the last record of the print file.
b. The core-address field contains the record number within the print file.

FORMAT OF PRINTER BACKUP FILE ON TAPE

A PBT file (a printer-backup file on tape) has the name PBTMCP/BACK-UP, may contain more than one printer file, and may span more than one reel.

FORMAT OF PRINTER BACKUP FILE ON DISK

Each PBD file (a printer-backup file on disk) has up to twenty 900-segment areas. The name of a PBD file is PBD/nnnnrrr, where nnnn is a serial number (in BCL) corresponding to the print file (which is incremented when a print file is opened on PBD), and rrr is the serial number of the backup file within the print file (analogous to reel number on a tape file). Thus, each print file may be composed of more than one physical backup file on disk, all with the same nnnn part.

FILE OPENING ACTION

When a print file is to be opened, the following action occurs:

a. If the file may go to a printer, the printers are checked for availability and one is used if possible.
b. If the file may go to a PBT (if an existing PBT is available), it is used; otherwise, if tape is available, a PBT is created and used.
c. If the file may go to PBD, a PBD is created and used.
d. If a unit is not found for the file, a message is typed to inform the operator. If a unit of the specified type is made available, it is used. If the operator changes the type with an OU reply, the above process is repeated.

SPECIAL FORMS

If the print file is opened on a printer-backup file, any special forms requirement is deferred until the backup file is printed. If the print file is opened on a printer:

a. A printer is chosen.
b. The operator is informed that special forms are required on that unit by the message # <unit> FM RQD. . . . The operator may then:
   1) Load the forms onto that unit and reply OK.
   2) Load the forms onto the other printer, SV the first printer, and reply OU LP.
   3) Reply OU MT or OU DK to force the chosen printer to be released to open a backup file.

PRINTED IN U.S. AMERICA 10-67 1031986

When a backup is printed which requires special forms, the message # FM RQD <unit> FOR <mfid> / <fid> OF <program name> will be typed, to which the operator may reply with OK, WY, or DS.

CLOSING A PRINT FILE ON DISK

When a print file on disk is closed, if the system option autoprint is set, it is scheduled to be printed. If autoprint is not set, a message is typed to inform the operator that a PBD exists and may be printed by the message PBD nnnn REL... .

LOGGING OF PB FILES

When a print file is printed from a PB file, an entry is made into the log containing the header card information of the program which initially created the print file, and all other appropriate information. The code (in the first word of "General Program Information") is 5, to indicate the printing of a print backup file.

SECTION 7
CHARACTER AND WORD MODE OPERATORS

OPERATOR DESCRIPTION

This section gives a shorthand version of the action of each Operator. It is written in such a form to give the programmer a quick description of each Operator and display the contents of the key flip-flops after the Operator has been executed.

Legend of the symbols used:

@	-	ADDRESS OF or ADDRESSED BY; this indicates the register which has memory addresses for the Operator. Example, A ← @ M, this indicates that the A register is loaded from memory by the cell addressed by the M register.
A, B, C etc.	-	This always indicates the register that will be found on the display panel for the Processor.
xx	-	Repeat Count Field, which is the two left octades of the T register.
←	-	REPLACEMENT ARROW; this indicates the contents of the register to the left of the arrow is replaced by the contents in the register on the right of the arrow.
⇒	-	INCLUSIVE; this indicates that all numbers between the two numbers shown are included. Example; 4 ⇒ 1 indicates 4, 3, 2, 1.
[ ]	-	FIELD DEFINITION; the numbers within the brackets are the bits of the register which define the field used. Example; A [15 ⇒ 1] means the first bit through the 15th bit of the A register.
>	-	Greater than
<	-	Less than
=	-	Equal to
≠	-	Not equal to
≥	-	Greater than or equal to
CS	-	Control State Operator

PRINTED IN U.S. AMERICA 10-67 1031986

WORD MODE OPERATORS

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
LS45	VFIL	Variable Field Isolate	1	-	-	-		"L" length field right justified "S" bits. All other bits ← 0	1	-	1.34.0
x051 ┐ x451 ┘	ZFNL	Br. Fwd. Non Destructive	1	1	-	-		If x plus bit 9 ≠ 0, then initiate syllable branch operator; (forward)	1	-	1.33.0
x151 ┐ x551 ┘	ZBNL	Br. Bkwd. Non Destructive	1	1	-	-		If x plus bit 9 ≠ 0, then initiate syllable branch operator; (backward)	1	-	1.33.0
x251 ┐ x651 ┘	ZFDL	Br. Fwd. Destructive	1	1	-	-		If x plus bit 9 ≠ 0, then initiate syllable branch operator; (forward); A ← 0	0	0	1.33.0
x351 ┐ x751 ┘	ZBDL	Br. Bkwd. Destructive	1	1	-	-		If x plus bit 9 ≠ 0, then initiate syllable branch operator; (backward); A ← 0	0	0	1.33.0
0051	DELL	Delete	-	-	-	-		AROF ← 0	0	-	1.33.0
0055	NOOP	No Operation	-	-	-	-	-	L← (L + 1)	-	-	1.19.0
XX55	DIAL	Dial A	1	-	-	-	-	G← T (12 ⇒ 10), H← T (9 ⇒ 7)	1	-	1.19.0
0061	VARL	Set Variant	-	-	-	-	-	VARF← 1, SALF ←0	-	-	1.19.0
XX61	DIBL	Dial B	-	1	-	-	-	K← T (12 ⇒ 10), V← T (9 ⇒ 7)	-	1	1.19.0
XX65	TRFL	Transfer Bits	1	1	-	-	-	B[K,V]←XX bits from A[G,H]	0	1	1.20.0
XX71	CFLL	Compare Field Low	1	1	-	-	-	If XX bits in B[K,V]<XX bits in A[G,H] then A(48 ⇒ 2)← 0, A01F←1. Else A(48 ⇒ 1)←0	1	1	1.22.0

197

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
XX75	CFEL	Compare Field Equal	1	1	-	-	-	If XX bits in A[G,H] = XX bits in B[K,V] then A(48 ⇒ 2)← 0, A01F← 1. Else A(48 ⇒ 1)←0	1	1	1.21.0
0105	AD2L	Double Precision Add	1	1	-	-	-	$B_1 B_2 \leftarrow A_1 A_2 + B_1 B_2$	1	1	1.19.0
0101	AD1L	Single Precision Add	1	1	-	-	-	B←A + B	0	1	1.01.0
0111	PREL	Program Release	1	-	48	$\overline{46}$		Presence Bit Interrupt	1	-	3.04.0
			1	-	48	46		A← @A[15 ⇒ 1]; "NCSF=0" A46← 0; @M←A	0		
					48	-		A←@(A Relative); "NCSF=1 If A28 cont. bit Int. Set, Else Prog. Rel. Int. Set; R+11← @(M).			
0115	LONL	Logical Negate	1	-	-	-	-	A [1 ⇒ 47] ← Δ A [1 ⇒ 47] ;	1	-	1.12.0
0121	CSDL	Cond. Integer Store Dest.	1	1	48	$\overline{46}$	-	Presence Bit Interrupt	1	1	1.18.0
			1	1	48	46	29	B← Integer; @(A[15 ⇒ 1]) ← B	0	0	
			1	1	48	46	$\overline{29}$	@(A[15 ⇒ 1])← B	0	0	
			1	1	$\overline{48}$		29	B ←Integer; @(A Relative) ← B	0	0	
			1	1	$\overline{48}$		$\overline{29}$	@(A Relative) ←B	0	0	

198

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

Octal Code	Mne. Name	Operator	Stack Condition AROF	BROF	Control Bits			Operator Action	Stack After Operator AROF	BROF	Flow Chart
0125	BGEL	"B" Greater than or Equal to "A"	1	1	-	-	-	B ← 0; If B ≥ A then B01 ← 1;	0	1	1.13.0
0131	BBCL	Branch Backward Conditional	1	1	-	-	B01	Continue to next syllable in sequence	0	0	1.15.0
			1	1	48	$\overline{46}$	-	Presence Bit Interrupt	1	1	
			1	1	48	46	$\overline{B01}$	C ← A [15 ⇒ 1] ; L ← 0;	0	0	
			1	1	$\overline{48}$	-	$\overline{B01}$	C ← C - A [03 ⇒ 12] ; L ← L - A [01 ⇒ 02] with possible decrement to "C"	0	0	
0135	RJPL	Branch Return						Refer to Subroutine operators Page 7-13.			1.16.0
0141	INDL	Index	1	1	-	-	-	A [15 ⇒ 1] ← A [15 ⇒ 1] + B [15 ⇒ 1] .	1	0	1.31.0
CS 0211	IINL	Interrogate Interrupt	-	-				If interrupt set then C← IAR, L← 0, S←100, IAR←0, Else NOOP.	-	-	3.03.0
0215	LOOL	Logical "OR"	1	1	-	-	-	A ←Results of Logically "ORing" the bits of the "A" Reg. with "B" Reg.; A48 ←B48.	1	0	1.09.0

199

Octal Code	Mne. Name	Operator	Stack Condition AROF	BROF	Control Bits			Operator Action	Stack After Operator AROF	BROF	Flow Chart
0221	CSNL	Cond. Integer Store Non. Dest.	1	1	48	$\overline{46}$		Presence Bit Interrupt	1	1	1.18.0
			1	1	48	46	29	B←Integer; @(A 15 ⇒ 1]) ←B 1]);	0	0	
			1	1	48	46	$\overline{29}$	@(A[15 ⇒ 1]) ←B	0	0	
			1	1	$\overline{48}$		29	B←Integer; @(A Relative)←B	0	0	
					$\overline{48}$		$\overline{29}$	@(A Relative)←B			
0225	BGAL	"B" Greater than "A"	1	1	-	-	-	B ← 0; If B > A then B01 ← 1;	0	0	1.13.0
0231	BFCL	Branch Forward Conditional	1	1	-	-	B01	Continue to next syllable in sequence	0	0	1.15.0
			1	1	48	$\overline{46}$		Presence Bit Interrupt	1	1	
			1	1	48	46	$\overline{B01}$	C ← A [15 ⇒ 1] ; L ← 0;	0	0	
			1	1	$\overline{48}$		$\overline{B01}$	C ← C + A [03 ⇒ 12] ; L ← L + [A01 ⇒ A02] ; With possible overflow to "C";	0	0	
0235	RNML	Return Normal						Refer to Subroutine Operators Page 7-9.	0	0	1.27.0
0241	MDVL	Construct Operand Call	1	1	-	-	-	B ← A; A ← B; A48 ← 1; Enter "OCSL" Syllable at Entry "A"	-	-	1.32.0
0301	SU1L	Single Prec. Subtract	1	1				B ← B - A	0	1	1.01.0
0305	SU2L	Double Precision Subtract	1	1	-	-	-	$B_1 B_2 \leftarrow B_1 B_2 - A_1 A_2$	1	1	1.02.0
0401	MU1L	Single Precision Multiply	1	1	-	-	-	B ← B X A;	0	1	1.03.0

PRINTED IN U.S. AMERICA 10-67

200

1031966

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
0405	MU2L	Double Precision Multiply	1	1	-	-	-	$B_1\ B_2 \leftarrow B_1\ B_2 \times A_1\ A_2$	1	1	1.04.0
CS 0411	RDTL	Read Timer	0	-				A[6 ⇒ 1] ← CC[TM6F ⇒ TM1F] A7 ← CCI03F	1	-	3.02.0
0415	LOAL	Logical "AND"	1	1	-	-	-	A ← Results of Logically "ANDING" the bits of the "A" Reg. with the "B" Reg: A48 ← B48.	1	0	1.09.0
0421	BSDL	"B" Store Destructive	1	1	48	$\overline{46}$	-	Presence Bit Interrupt	1	1	1.17.0
			1	1	48	46	-	@ A [15 ⇒ 1] ← B	0	0	
			1	1	$\overline{48}$	-	-	@ (A Relative) ← B	0	0	
0425	BNEL	"B" Not Equal to "A"	1	1	-	-	-	B ← 0; If B ≠ A then B01 ← 1;	0	1	1.13.0
0431	MSNL	Set Sign Bit	1	-	-	-	-	A47 ← 1;	1	-	1.25.0
0441	MSOL	Mark Stack						Refer to Subroutine Operators, Page 7-9.			1.26.0
1001	DV1L	Single Precision Divide	1	1	-	-	-	B ← B/A	0	1	1.05.0
1005	DV2L	Double Precision Divide	1	1	-	-	-	$B_1\ B_2 \leftarrow B_1\ B_2 / A_1\ A_2$	1	1	1.06.0
1011	COML	Communicate	-	-				@(R+11) ← Top of Stack Communicate Interrupt ← 1	-	-	3.01.0

201

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
1015	LOEL	Logical Equivalence	1	1	-	-	-	B Bit ← 1 If corresponding bits of B and A are same; B48 is not altered;	0	1	1.11.0
1021	BSNL	"B" Store Non Destructive	1	1	48	$\overline{46}$	-	Presence Bit Interrupt	0	1	1.17.0
			1	1	48	46	-	@ (A [15 ⇒ 1]) ← B	0	1	
			1	1	$\overline{48}$	-	-	@ (A Relative) ← B	0	1	
1025	EXCL	Exchange	1	1	-	-	-	A ↔ B	1	1	1.28.0
1031	CSSL	Change Sign Bit	1	-	-	-	-	A47 ← Δ A47	1	0	1.25.0
1235	RSPL	Return Special						Refer to Subroutine Operators Page 7-9.			1.27.0
1241	MDAL	Construct Descriptor Call	1	1	-	-	-	A ↔ B; A48 ← 1; Enter "DCSL" syllable at Entry "A"	-	-	1.32.0
1425	FCXL	Transfer F Field to C Field	1	1	-	-		B [15 ⇒ 1] ← A [30 ⇒ 16]	0	1	1.28.0
2015	RFBL	Reset Flag Bit	1	-	-	-	-	A48 ←	1	-	1.23.0
2021	LODL	Load	1	-	48	$\overline{46}$	-	Presence Bit Interrupt	1	-	1.30.0
			1	-	48	46	-	A ← @ A [15 ⇒ 1]	1	-	
			1	-	$\overline{48}$	-	-	A ← @ (A[10 ⇒])	1	-	
2025	DUPL	Duplicate	1	0	-	-	-	B ← A; BROF ←	1	1	1.29.0

202

PRINTED IN U.S.A. AMERICA 10-67
1031966

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
2031	TFBL	Test Flag Bit	0	1	-	-	-	A ← 0; If B48 = 0 then A01 ← 1;	1	1	1.24.0
CS											
2111	IORL	I/O Release	1	-	48	$\overline{46}$	-	Presence Bit Interrupt	1	-	3.04.0
			1	-	48	46	-	A ← @A[15 ⇒ 1] A46 ← 1, @M ← A	0	-	
			1	-	$\overline{48}$	-	-	A ← @(A Relative) A46 ← 1, @M ← A	0	-	
2131	JBCL	Word Branch Bkwd. Cond.	1	1	-	-	B01	Continue to next SYLL. in sequence	0	0	1.15.0
			1	1	$\overline{A48}$	-	$\overline{B01}$	C ← (C-A[10 ⇒ 1]), L ← 0	0	0	
			1	1	A48	A46	$\overline{B01}$	C ← A[15 ⇒ 1], L ← 0	0	0	
			1	1	A48	$\overline{A46}$	-	Presence Bit Interrupt	1	1	
2141	FXSL	F & S Reg. Set/Store	1	1	$\overline{A02}$	$\overline{A01}$	-	B [30 ⇒ 16] ← F	0	1	1.31.0
			1	1	$\overline{A02}$	A01	-	B [15 ⇒ 1] ← S	0	1	1.31.0
			1	1	A02	$\overline{A01}$	-	SALF ← 1; F ← B[30 ⇒ 16]	0	0	1.31.0
			1	1	A02	A01	-	S ← B[15 ⇒ 1]	0	0	1.31.0
CS											
2211	HP2L	Halt P2	-	-	-	-	-	HP2F ← 1. P2 Executes SFIL.	-	-	3.05.0
2231	JFCL	Word Branch Fwd. Cond.	1	1	-	-	B01	Cont. to next SYLL. in sequence	0	0	1.15.0
			1	1	$\overline{A48}$	-	$\overline{B01}$	C ← (C + A[10 ⇒ 1]) L ← 0	0	0	
			1	1	A48	A46	$\overline{B01}$	C ← A[15 ⇒ 1], L ← 0	0	0	
			1	1	A48	$\overline{A46}$	-	Presence Bit Interrupt	1	1	

203

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
2411	CHPL	Conditional Halt	-	-	-	-	-	If stop operator SW. on, Stop Clock to Processor else NOOP	-	-	3.09.0
2431	IPSL	Interrogate Peripheral Status	0	-	-	-		Interrogate Peripheral Status Lines from CC. (See page 3-8.)	1	1	1.24.0
2541	LLLL	Link List Look Up	1	1	-	-		M ← B [15 ⇒ 1] ; If B ≥ A then A [47 & 45 ⇒ 16] ← 0, A [15 ⇒ 1] ← M; A [48 & 46] ← 1, else M ← B [15 ⇒ 1] , Load B @ M, Repeat Comparison	1	-	1.31.0
3001	DV3L	Integer Divide	1	1	-	-	-	B ← B/A until Exponent = 0;	0	1	1.07.0
3425	FFXL	Transfer F Field to F Field	1	1	-	-		B [30 ⇒ 16] ← A [30 ⇒ 16]	0	1	1.28.0
4015	SFBL	Set Flag Bit	1	-	-	-	-	A48 ← 1;	1	-	1.23.0
CS											
4111	IN1L	Initiate P1	0	1	-	-	-	Distribute INCW; Dist. IRCW; Dist. ICW; NCSF ← 1. If char. mode dist. ILCW. Ref. Pg. 2.9.	-	-	3.07.0
4121	ISDL	Integer Store Destructive	1	1	48	$\overline{46}$	-	Presence Bit Interrupt	1	1	1.18.0
			1	1	48	46	-	B made Integer; @ (A [15 ⇒ 1]) ← B	0	0	
			1	1	$\overline{48}$	-	-	B made Interger; @ (A Relative) ← B	0	0	
4125	BLEL	"B" Less Than or Equal to "A"	1	1	-	-	-	B ← 0; If B ≤ A then B01 ← 1;	0	1	1.13.0

204

PRINTED IN U.S. AMERICA 10-67
1081986

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
4131	BBUL	Branch Backward Unconditional	1	-	48	$\overline{46}$	-	Presence Bit Interrupt	1	-	1.14.0
			1	-	48	46	-	C ← A [15 ⇒ 1] ; L ← 0;	0	-	
			1	-	$\overline{48}$	-	-	C ← C - A [03 ⇒ 12] ; L ← L - A [01 ⇒ 02] with a possible Decrement of "C"	0	-	
CS 4211	PTOL	Initiate P2	0	0	-	-	-	M04F← 1; B← @M(INCW). Then same as INITIATE P1.			3.07.0
4221	ISNL	Integer Store Non Destructive	1	1	$\overline{48}$	$\overline{46}$	-	Presence Bit Interrupt	1	1	1.18.0
			1	1	48	46	-	B made Integer; @ (A [15⇒ 1])← B	0	1	
			1	1	$\overline{48}$	-	-	B made Integer; @ (A Relative)← B	0	1	
4225	BLAL	"B" Less Than "A"	1	1	-	-	-	B ← 0; If B < A Then B01 ← 1;	0	1	1.13.0
4231	BFUL	Branch Forward Uncondftional	1	-	48	$\overline{46}$	-	Presence Bit Interrupt	1	1	1.14.0
			1	-	48	46	-	C ← A [15 ⇒ 1] ; L ← 0;	0	-	
			1	-	$\overline{48}$	-	-	C ← C + A [03 ⇒ 12] ; L ← L + [A01 ⇒ A02] with possible overflow to "C"	0 0	- -	

205

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
CS 4411	IOOL	Initiate I/O	1	-	-	-	-	M04F ← 1; @M← A; C.C. Now initiates I/O using desc. in cell 10.	0	-	3.08.0
4425	BEQL	"B" Equal to "A"	1	1	-	-	-	B ← 0; If B = A Then B01F ← 1;	0	1	1.13.0
4431	MSPL	Reset Sign Bit	1	-	-	-	-	A47 ← 0;	1	-	1.25.0
4441	ECML	Enter Character Mode in Line	0	0	-	-		Construct RCW; Enter RDAL (xx04) at J = 2	-	0	1.26.0
CS 5111	IFTL	Initiate For Test	0	1	-	-	-	Distribute INCW and set up TM register; dist. IRCW; dist. ICW; dist. ILCW; load A • B; J← TM[4 ⇒ 1]; NCSF← TM5F; CCCF ← TM6F.	-	-	3.07.0
5425	CCXL	Transfer C Field to C Field	1	1	-	-		B [15 ⇒ 1] ← A [15 ⇒ 1]	0	1	1.28.0
6131	JBUL	Word Br. Bdwd. Uncond.	1	-	48	$\overline{46}$	-	Presence Bit Interrupt	1	-	1.14.0
			1	-	48	46	-	C←A[15 ⇒ 1]; L← 0	0	-	
			1	-	$\overline{48}$	-	-	C-(A[10 ⇒ 1]); L←0	0	-	
6231	JFUL	Word Br. Fwd. Uncond.	1	-	48	$\overline{46}$	-	Presence Bit Interrupt	1	-	1.14.0
			1	-	48	46	-	C ← A[15 ⇒ 1]; L← 0	0	-	
			1	-	$\overline{48}$	-	-	C+(A[10 ⇒ 1]); L←0	0	-	

206

PRINTED IN U.S. AMERICA 10-67

1031966

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
6431	TIOL	Interrogate I/O Channels	0	-	-	-		Test for which I/O Channel is busy; A [03 ⇒ 01] ← [1 - I/01; 2 - I/02; 3 - I/03; 4 - I/04];	1	-	1.24.0
7001	DV4L	Remainder Divide	1	1	-	-	-	B ← Remainder of Integer Divide of B/A;	0	1	1.08.0
7031	SSFL	Search For Flag Bit	1	-	$\overline{48}$ 48	-		M + 1, Load A @ M; A48 & A46 ← 1 A47 ← 0, A [45 ⇒ 16] ← 0; A [15 ⇒ 1] ← M	1	-	1.25.0
7425	CFXL	Transfer C Field to F Field	1	1	-	-		B [30 ⇒ 16] ← A [15 ⇒ 1]	0	1	1.28.0

207

WORD MODE OPERATORS - SUBROUTINE OPERATORS

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits			Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
0135	RJPL	Branch Return	1	0	48	$\overline{46}$		Presence Bit Interrupt	1	-	
			1	0	48	46		B ← A; C ← B [15 ⇒ 1] ; L, G, H, K, V ← 0; S ← A [16 ⇒ 30] B ← @S; F ← B [16 ⇒ 30] ; MSFF ← B31; SALF ← B32; S ← S - 1; R ← B [39 ⇒ 45] ;	0	0	1.16.0
0435	REWL	Exit	-	-				B ← @F; C ← B [15 ⇒ 1] ; L ← B [37 & 38] ; G ← B [36 ⇒ 34;] K ← B [33 ⇒ 31] ; H ← B [44 ⇒ 42] ; V ← B [41 ⇒ 39] ; S ← B [16 ⇒ 30] ; B ← @S; S ← S-1; F ← B [16 ⇒ 30] ; MSFF ← B31; SALF ← B32; R ← B [39 ⇒ 45] ;	0	0	3.11.0
0235	RNML	Return Normal	1					This instruction is similar to REWL with the following exceptions: 1. The "A" register is left valid through the execution of the operator. 2. At completion of the operator, the OCSL flow chart is entered at Point "A" if B46 of RCW is off and OCSL flow chart is entered at Point "A" if B46 of RCW is on.	1	0	1.27.0

208

PRINTED IN U.S. AMERICA 10-67

1031986

Octal Code	Mne. Name	Operator	Stack Condition AROF	Stack Condition BROF	Control Bits	Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
1235	RSPL	Return Special	1	-		This operator is similar to RNML except that the RCW is read from memory with the "S" register instead of with the "F" register.	1	0	1.27.0
xx00	RECL	Exit Character Mode	-	-		This operator is similar to REWL except CWMF is reset at completion of operator.	0	0	2.39.0
0441	MSOL	Mark Stack	-	-		Initially if either "A" reg. or "B" reg. is valid they are pushed into the stack and then a mark Stack Control Word is constructed as follows: B [16 ⇒ 30] ← F; B32 ← MSFF; B31 ← SALF; B[42 ⇒ 34] ← R; @S ← B; F ← S; If SALF = 1 and MSFF = 0 @ (R + 7) ← B MSFF ← 1;	0	0	1.26.0

602

CHARACTER MODE OPERATORS

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
xx00	RECL	Exit Character Mode	-	-	See Subroutine Operators Page 7-13	0	0	2.39.0
xx02	SBDL	Skip Bit Destination	-	-	(S, K, V) ← (S, K, V) + xx Bits	-	-	2.32.0
xx03	SBSL	Skip Bit Source	-	-	(M, G, H) ← (M, G, H) + xx Bits	-	-	2.31.0
xx04	RDAL	Recall Destination Address	B48	$\overline{B46}$	Presence Bit Interrupt	-	0	2.16.0
			B48	B46	S ← @ (F - xx [15 ⇒ 1]); K ← 0; V ← 0	-	0	
			$\overline{B48}$		S ← @ (F - xx [15 ⇒ 1]); K ← @ (F - xx [18 ⇒ 16]); V ← 0	-	0	
xx05	TWDL	Transfer Words	-	-	If (G, H) > 0 then M ← M + 1, G ← 0, H ← 0; If (K, V) > 0 then S ← S + 1, K ← 0, V ← 0; xx Words @ S ← @ M;	0	0	2.27.0
xx06	SDPL	Set Destination Address	-	-	S ← (F - xx); K ← 0; V ← 0	-	0	2.14.0
xx07	SDAL	Transfer Destination Address	-	-	If V > 0 then (S, K) ← (S, K) + 1; K ← 3 Bits @ (S, K); S ← 15 Bits @ (S, K); V ← 0	0	0	2.13.0
xx12	TBZL	Trans. Blk for Non-numeric	-	-	TFFF ← 1; If V > 0 then (S,K) ← (S,K)+1; V ← 0; If Char @ (S,K) is ≤ Zero then Char @ (S,K) ← 60, continue until xx Char. tested; else EXIT. TFFF ← 0 if Numeric Char. is found.	-	-	2.05.0

012

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
xx14	STDL	Store Destination Address	-	-	If V > 0 then (S, K) ← (S, K) + 1; V ← 0; @ (F - xx [15⇒1])←S; @ (F-xx [18 ⇒16])←K	-	0	2.15.0
xx15	STSL	Store Source Address	-	-	If H > 0 then (M, G) ← (M, G + 1); H ← 0; @ (F-xx [15 ➡ 1]) ←M; (F-xx [18⇒16])←G	0	-	2.11.0
xx16	FSDL	Skip Forward Destination	-	-	If V > 0 then (S, K) ← (S, K) + 1; V ← 0; (S, K) ← (S, K) + xx	-	-	2.25.0
xx17	RSDL	Skip Reverse Destination	-	-	If V > 0 then (S, K) ← (S, K) + 1; V ← 0; (S, K) ← (S, K) - xx	-	-	2.26.0
xx22	SSPL	Set Source Address	-	-	M← (F-xx); (G,H)← 0.	0	-	2.10.0
xx24	TEQL	Test For Equal	-	-	TFFF ← 0; If Char. @ (M, G) = xx then TFFF← 1	1	-	2.07.0
xx25	TNEL	Test For Not Equal	-	-	TFFF ← 0; If Char. @ (M, G) ≠ xx then TFFF ← 1	1	-	2.07.0
xx26	TGEL	Test For Greater or Equal	-	-	TFFF ← 0; If Char. @ (M, G) ≥ xx then TFFF ← 1	1	-	2.07.0
xx27	TGTL	Test For Greater	-	-	TFFF ← 0; If Char. @ (M, G) > xx then TFFF ← 1	1	-	2.07.0

211

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
xx30	RSSL	Skip Reverse Source	-	-	If H > 0 then (M, G) ← (M, G) + 1; H ← 0; (M, G) ← (M, G) - xx	-	-	2.24.0
xx31	FSSL	Skip Forward Source	-	-	If H > 0 then (M, G) ← (M, G) + 1; H ← 0; (M, G) ← (M, G) + xx	-	-	2.24.0
xx34	TLEL	Test For Equal or Less	-	-	TFFF ← 0; If Char. @ (M, G) ≤ xx then TFFF ← 1	1	-	2.07.0
xx35	TLTL	Test For Less	-	-	TFFF ← 0; If Char. @ (M, G) < xx then TFFF ← 1	1	-	2.07.0
xx36	TANL	Test For Alphanumeric	-	-	TFFF ← 0; If Char. @ (M, G) = Alpha or Numeric Then TFFF ← 1	1	-	2.08.0
xx37	TEBL	Test Bit	-	-	TFFF ← 0; If Bit @ (M, G, H) = T07F Then TFFF ← 1	1	-	2.30.0
xx40	INTL	Increase Tally	-	-	R ← R + xx	-	-	2.35.0
xx41	STAL	Store Tally	-	-	@ (F - xx [6 ➡ 1]) ← R	-	0	2.34.0
xx42	STEL	Set Tally	-	-	R ← xx	-	-	2.35.0
xx43	CLRL	Call Repeat Field	-		Refer to Special Character Mode Operators Page 7-18.			2.33.0
xx44	CJOL	Jump Out of Loop Conditional		$\overline{\text{TFFF}}$	X ← @ X [30 ⇒ 16] ; (C, L) ← (C, L) + xx			2.21.0
				TFFF	Continue in Seq.			

212

PRINTED IN U.S. AMERICA 10-67

1031396

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	BROF	Flow Chart
xx45	CFJL	Jump Forward Conditional		$\overline{TFFF}$	(C, L) ← (C, L) + xx			2.22.0
				TFFF	Continue in Seq.			
xx46	JOLL	Jump Out of Loop Uncond.	-	-	X ← @ X [30 ⇒ 16] ; (C, L) ← (C, L) + xx			2.21.0
xx47	FWJL	Jump Forward Unconditional	-	-	(C, L) ← (C, L) + xx			2.22.0
xx50	RPAL	Recall Control Address	B48	$\overline{B46}$	Presence Bit Interrupt	0	-	2.18.0
			B48	B46	C ← @ (F - xx [15 ⇒ 1]); L ← 0			
			$\overline{B48}$	B46	C ← @ (F - xx [15 ⇒ 1]); L ← @ (F - xx [38 ⇒ 37]); (C, L) ← (C, L) + 1	0	-	
xx51	ENLL	End Loop			If X [36 ⇒ 31] > 0 Then X [36 ⇒ 31] ← X [36 ⇒ 31] - 1, C ← X [15 ⇒ 1] , L ← X [38 ⇒ 37] ; Else X ← @ X [30 ⇒ 16]	0	-	2.20.0
xx52	BELL	Begin Loop	-	-	Refer to Special Character Mode Operators Page 7-18.			2.19.0
xx53	RSAL	Recall Source Address	B48	$\overline{B46}$	Presence Bit Interrupt	0	-	2.12.0
			B48	B46	M ← @ (F - xx [15 ⇒ 1]); G ← 0; H ← 0			
			$\overline{B48}$	B46	M ← @ (F - xx [15 ⇒ 1]); G ← (F - xx [18 ⇒ 16]); H ← 0			

213

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	BROF	Flow Chart
xx54	STPL	Store Control Address	-	-	@ (F - xx [15 ⇒ 1]); ← C; @ (F - xx [38 ⇒ 37]) ← L	0	-	2.17.0
xx55	CRJL	Jump Reverse Conditional		$\overline{TFFF}$	(C, L) ← (CL) - xx	-	-	2.23.0
				TFFF	Continue in Seq.			
xx56	SSAL	Transfer Source Address			G ← 3 Bits @ (M, G); M ← 15 Bits @ (M, G)	0	0	2.09.0
xx57	REJL	Jump Reverse Unconditional			(C, L) ← (C, L) - xx	-	-	2.23.0
xx60	SEQL	Compare Equal	-	-	TFFF ← 0; If xx Char. @ (M, G) = @ (S, K), Then TFFF ← 1	1	1	2.06.0
xx61	SNEL	Compare Not Equal			TFFF ← 0; If xx Char. @ (M, G) ≠ @ (S, K), Then TFFF ← 1	1	1	2.06.0
xx62	SGEL	Compare Greater or Equal			TFFF ← 0; If xx Char. @ (M, G) ≥ @ (S, K) Then TFFF ← 1	1	1	2.06.0
xx63	SGTL	Compare for Greater			TFFF ← 0; if xx Char @(M,G) > @(S,K) then TFFF ← 1.	1	1	2.06.0
xx64	SEBL	Set Bit			xx Bits @ (S, K, V) ← 1	-	1	2.36.0
xx65	REBL	Reset Bit			xx Bits @ (S, K, V) ← 0	-	1	2.29.0

214

PRINTED IN U.S. AMERICA 10-67
1031986

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
xx66	OCOL	Output Convert	-	-	If (G, H) > 0 Then M ← M + 1, G ← 0, H ← 0; If V > 0 Then K ← K + 1, V ← 0; TFFF ←1; xx Char @ (S, K) ←Decimal Equivalent of @ M; If Overflow occurs then TFFF ←0	0	-	2.38.0
xx67	ICOL	Input Convert			If (H, G) > 0 Then M ← M + 1; G ← 0, H ← 0. If V > 0 Then K ← K + 1, V ← 0; xx Char @ (S, K) ← Octal Equivalent of @ M;	0	-	2.37.0
xx70	SLEL	Compare Equal or Less			TFFF ← 0; If xx Char. @ (M, G) ≤ @ (S, K) Then TFFF ← 1	1	1	02.06.0
xx71	SLTL	Compare Less			TFFF ← 0; If xx Char. @ (M, G) < @ (S, K) Then TFFF ← 1	1	1	02.06.0
xx72	FSUL	Field Subtract			TFFF ← 0; xx Char. @ (S, K) ← @ (S, K) Algebraically subtracted from @ (M, G); If Overflow occurs Then TFFF ← 1	1	1	02.28.0
xx73	FADL	Field Add	-	-	TFFF ← 0; xx Char. @ (S, K) ← @ (S, K) Algebraically Added to @ (M, G); If Overflow occurs Then TFFF ← 1	1	1	02.28.0
xx74	TPDL	Transfer Program Characters	-	T07F	(C, L)← (C, L) + 1 Char. And Continue	-	1	2.05.0
				$\overline{T07F}$	xx Char. @ (S, K) ← @ (C, L)			

215

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
xx75	TNDL	Transfer Numerics			TFFF ← 0; xx Numeric Bits of Char. @ (S, K) ← @ (M, G) Zone Bits ← 0; If @ (MG) = Minus Then TFFF ⇒ 1	1	1	2.04.0
xx76	TZDL	Transfer Zones			xx Zone Bits of Char. @ (S, K) ← @ (M, G)	1	1	2.03.0
xx77	TSDL	Transfer Source Char.			xx Char. @ (S, K) ← @ (M, G)	1	1	2.02.0
0100	ILEL	In Line Exit Char. Mode	-	-	Similar to RECL (xx00)	1	1	2.39.0
3411	SFTL	Store For Test	-	-	Similar to SFIL (3011) except access cell 0	-	-	3.06.0
5111	IFTL	Initiate For Test	-	-	Similar to INIL (4111)	1	1	3.07.0

216

PRINTED IN U.S. AMERICA 1047
1031986

SPECIAL CHARACTER MODE OPERATORS

Octal Code	Mne. Name	Operator	Control Bits		Operator Action	Stack After Operator AROF	Stack After Operator BROF	Flow Chart
xx43	CLRL	Call Repeat Field	-	-	If @F-xx [6⇒1] ≠ 0, then (T [12⇒7] @C&L+1) ← @F-xx [6⇒1]; else Branch forward unconditionally T [12⇒7] of next syllable.	0	-	2.33.0
xx52	BELL	Begin Loop	-	-	@X [30⇒16] ← X; X [30⇒16] ← X [30⇒16] + 1; X [36⇒31] ← T [12⇒7] - 1; X [15⇒1] & X [38⇒37] ← C&L+1; T←C&L+1;	0	-	2.19.0

SECTION 8
INDEX OF DA'S AND PRINTS

INDEX OF DA'S AND PRINTS

PROCESSOR

A Register	A Rack	65.10.nn.0
B Register	B Rack	65.66.nn.0
C Register	J Rack	65.38.nn.0
E Register	D Rack	65.70.nn.0
F Register	J Rack	65.14.nn.0
G Register	D Rack	65.54.nn.0
H Register	D Rack	65.58.nn.0
I Register	D Rack	65.88.nn.0
J Register	E Rack	65.78.nn.0
K Register	D Rack	65.22.nn.0
L Register	D Rack	65.42.nn.0
M Register	J Rack	65.62.nn.0
N Register	D Rack	65.30.nn.0
P Register	E Rack	65.46.nn.0
R Register	J Rack	65.74.nn.0
S Register	J Rack	65.34.nn.0
T Register	E Rack	65.50.nn.0
V Register	D Rack	65.26.nn.0
X Register	A Rack	65.18.nn.0
Y Register	D Rack	65.82.nn.0
Z Register	D Rack	65.86.nn.0

AROF	A Rack	65.10.49.2
BROF	A Rack	65.66.49.2
CWMF	E Rack	65.38.16.0
HLTF	E Rack	65.90.11.0
MRAF	E Rack	65.62.16.0
MROF	E Rack	65.62.16.0
MWOF	E Rack	65.62.17.0
NCSF	D Rack	65.46.49.0
PROF	E Rack	65.46.49.1
Q01F	A Rack	65.90.01.0
Q02F	E Rack	65.90.02.2

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

PROCESSOR (continued)

Q03F	D Rack	65.90.03.0
Q04F	E Rack	65.90.04.0
Q05F	D Rack	65.90.05.0
Q06F	D Rack	65.90.06.0
Q07F	D Rack	65.90.06.1
Q08F	B Rack	65.90.07.0
Q09F	B Rack	65.90.08.0
Q12F	D Rack	65.90.09.0
SALF	A Rack	65.90.10.0
TROF	E Rack	65.50.15.0

CENTRAL CONTROL

Rack A

Crosspoint Logic 55.10.00.0 - 55.17.01.0
Memory Read Exchange

Rack B

Memory Write Exchange

Rack D

I/O Non-Tape Exchange
Peripheral Designation 55.50.16.1 - 55.50.31.1

Rack E

Incandescent Drivers	
System Clock	55.00.12.0
Load Flip Flop	55.00.11.0
Interrupt Control	
I/O Selection	55.00.30.0
Interrupt Address Registers	55.00.66.1 - 55.00.63.1
I/O Tape Exchange	
Real Time Clock	55.00.67.0 & 55.00.68.0

I/O

Registers	D.A.		D.A.
D Register	60.04.14.1	OB Register	60.05.01.0
IB Register	60.05.05.0	W Register	60.04.01.0
IR Register	60.05.09.0	WB Register	60.05.07.0
LP Register	60.01.36.0		

Counters	D.A.
CC	60.01.04.0
PC	60.01.47.0
SC	60.01.53.0

Flip Flops	D.A.		D.A.
AOFF	60.01.02.0	MANF	60.01.41.0
BKWF	60.03.04.0	MAOF	60.01.41.0
EXNF	60.01.24.0	OBCF	60.05.01.0
FWDF	60.03.04.0	PUCF	60.03.14.0
HOLF	60.01.25.0	RCNF	60.03.16.0
IMCF	60.09.02.0	RECF	60.09.02.0
IMFF	60.03.07.0	REMF	60.09.02.0
IMIF	60.03.07.0	SKFF	60.03.15.0
LCHF	60.01.34.0	SHOF	60.03.15.0
LPWF	60.01.39.0	STRF	60.01.27.0
		VRCF	60.03.16.0

B 460 CORE MEMORY

	Sheet Location	TFR Page
MAR 1 - 6	1	57
MAR 7 - 12	2	57A
Low X Write Drivers Read Switches and Bi-Di	3	58
High X Read Drivers Write Switches and Bi-Di	4	58A
Low Y Write Drivers Read Switches and Bi-Di	5	59
High Y Read Drivers Write Switches and Bi-Di	6	59A
MIR 1 - 6	7	60
MIR 7 - 12	8	60A
MIR 13 - 18	9	61
MIR 19 - 24	10	61A
MIR 25 - 30	11	62
MIR 31 - 36	12	62A
MIR 37 - 42	13	63
MIR 43 - 48	14	63A
CT Counter, MPEF & MIR 49	15	64
Parity Error EV-1 - EV-8	16	64A
Parity Error EV-9 - PERL	17	65
Memory Timing Circuits, MIRC and Clock Driver	18	65A
MIR Test Error A and B	19	66
MIR Test Error C and D	20	66A
Checkerboard and Test Circuits	21	67
Neon Drivers MIR 1 - 24	22	67A
Neon Drivers MIR 25 - 48	23	68
Neon Drivers MPEF - MAR12	24	68A
Manual Switches	25	69
Input Lines	26	69A

PRINTED IN U.S. AMERICA 10-67

1031986

B 460 CORE MEMORY (continued)

	Sheet Location	TFR Page
Output Lines	27	70
Core Stack Terminations	28	70A
Timing Diagram	29	54
Core Memory Logic Diagram		71-72

SECTION 9
MAINTENANCE INFORMATION

CLOCK TEST POINTS

PROCESSOR	I/O	CENTRAL CONTROL	MEMORY
AA C6 B7	A C7 B7	AA B6 P7	P33D-4
AC C6 B7	C B8 P7	EA B8 P7	AAC7B7
AE C6 B7	E C7 B7		
BA C7 B7			
BC C7 B7			
BE C6 B7			
JC C7 B7			
JE C7 B7			
DA C7 B7			
DC C7 B7			
DE C7 B7			
EA B7 P7			
EE B7 P7			

DRBL TEST POINTS

Processor A	EAC3C0
Processor B	EAC3R0
Lore & C.C.	EAD3C0
I/O	EAD3R0

NOTE: Additional Output Pins

B7 ≡ B7 E7 K7
P7 ≡ P7 T7 X7
C0 ≡ C0 E0 H0 K0
R0 ≡ R0 T0 V0 X0
4 ≡ 4 Thru 12

PROCESSOR MAINTENANCE PANEL TOGGLE SWITCHES

SWITCH NUMBER	ACTION
US01X	INHIBIT COUNT REPEAT FIELD (T) (BY -1, -4, -8)
US02X	INHIBIT COUNT G & H
US03X	INHIBIT COUNT M
US04X	INHIBIT COUNT K, V & N
US05X	INHIBIT COUNT S
US06X	INHIBIT COUNT C
US07X	INHIBIT COUNT L
US08X	INHIBIT RESET AROF
US09X	INHIBIT RESET BROF
US10X	INHIBIT T ← P @ (L)
US11X	INHIBIT STORE
US12X	INHIBIT I/O REQUESTS (IOOL)

PRINTED IN U.S. AMERICA 10-67

1031986

SWITCH NUMBER	ACTION
US13X	INHIBIT INTERRUPTS
US14X	STOP INSTRUCTION (OPERATOR)(CHPL)
US15X	"STOP CLOCK" - DRIVER (PROCESSOR ONLY)
US16X	STOP ON SECL
US17X	STOP ON INTERRUPT (PROCESSOR TYPE)
US18X	STOP WHEN NORMALIZED
US19X	STOP ON J COUNT (USED WITH US20X - US23X)
US20X ⇒ US23X	J CODE SWITCHES
US24X	SINGLE PULSE
US25X	SINGLE PULSE MEMORY WRITE
US26X	INHIBIT RESET "A" MANTISSA
US27X	INHIBIT 42 BIT ADDER (B02ZD)
US28X	LOCK UP ON J COUNT = US20X - US23X SETTING
US29X	MEMORY LOAD (USE WITH US25X)

T REGISTER DECODING - WORD MODE

T REGISTER FLIP FLOP		SUB CLASS 12	11	10	9	8	7	CLASS 6	5	4	3	TYPE 2	1	
SINGLE PRECISION ADD	AD1L					0	1	0	0	0	0	0	1	*
SINGLE PRE. SUBTRACT	SU1L					1	1	0	0	0	0	0	1	
SINGLE PRE. MULTIPLY	MU1L				1			0	0	0	0	0	1	
SINGLE PRE. DIVIDE	DV1L		0	1				0	0	0	0	0	1	
INTEGER DIVIDE	DV3L	0	1	1				0	0	0	0	0	1	
REMAINDER DIVIDE	DV4L	1	1	1				0	0	0	0	0	1	
DOUBLE PRE. ADD	AD2L					0	1	0	0	0	1	0	1	*
DOUBLE PRE. SUBTRACT	SD2L					1	1	0	0	0	1	0	1	
DOUBLE PRE. MULTIPLY	MU2L				1			0	0	0	1	0	1	
DOUBLE PRE. DIVIDE	DV2L			1				0	0	0	1	0	1	
COMMUNICATE	COML	0	0	1				0	0	1	0	0	1	
READ TIMER	RDTL	0	0		1			0	0	1	0	0	1	
INTERROGATE INTERRUPT	IINL	0	0			1		0	0	1	0	0	1	
PROGRAM RELEASE	PREL	0	0				1	0	0	1	0	0	1	PIRL
INPUT RELEASE	IORL	0	1				1	0	0	1	0	0	1	PIRL
HALT PROCESSOR 2	HP2L	0	1			1		0	0	1	0	0	1	
CONDITIONAL HALT	CHPL	0	1		1			0	0	1	0	0	1	

T REGISTER FLIP FLOP		SUB CLASS 12	11	10	9	8	7	CLASS 6	5	4	3	TYPE 2	1	
STORE FOR INTERRUPT	SFIL	0	1	1				0	0	1	0	0	1	SFIL
STORE FOR TEST	SFTL	0	1	1	1			0	0	1	0	0	1	SFIL
INITIATE	INIL	1	0			1		0	0	1	0	0	1	INIL
INITIATE FOR TEST	IFTL	1	0	1		1		0	0	1	0	0	1	INIL
INITIATE PROCESSOR 2	PTOL	1	0		1			0	0	1	0	0	1	PIOL
INITIATE IN/OUT	IOOL	1	0		1			0	0	1	0	0	1	PIOL
NOT ASSIGNED		1	0	1				0	0	1	0	0	1	
		1	1			1		0	0	1	0	0	1	
		1	1		1			0	0	1	0	0	1	
		1	1	1				0	0	1	0	0	1	
		1	1	1				0	0	1	0	0	1	
* COMMON LEVELS ARITHMETIC OPERATORS	T01L	x	x	x	x	x	x	0	0	0	x	0	1	
	T02L	x	x	x	x	x	x	0	0	0	0	0	1	
	T03L	x	x	x	x	x	x	0	0	0	1	0	1	
	T04L	x	x	x	x	x	1	0	0	0	0	0	1	
	T05L	x	x	x	x	x	1	0	0	0	1	0	1	
	T06L	x	x	x	1	x	x	0	0	0	x	0	1	
	T07L	x	x	1	x	x	x	0	0	0	x	0	1	
	T08L	x	1	x	x	x	x	0	0	0	x	0	1	
	T11L	x	x	1	x	x	x	0	0	0	0	0	1	
	T12L	x	0	1	x	x	x	0	0	0	x	0	1	
C03L — T50L	LONL						1	0	0	1	1	0	1	
C03L — T50L	LOOL					1		0	0	1	1	0	1	
C03L — T50L	LOAL				1			0	0	1	1	0	1	
C03L — T50L	LOEL			1				0	0	1	1	0	1	
C03L — T55L	RFBL		1					0	0	1	1	0	1	
C03L — T55L	SFBL	1						0	0	1	1	0	1	
C04L — T51L	BSDL				1			0	1	0	0	0	1	
C04L — T51L	BSNL			1				0	1	0	0	0	1	
C04L — T52L (T52L')	ISDL	1					1	0	1	0	0	0	1	
C04L — T52L (T52L')	ISNL	1				1		0	1	0	0	0	1	
C04L — T52L (T52L')	CSDL	0					1	0	1	0	0	0	1	
C04L — T52L (T52L')	CSNL	0				1		0	1	0	0	0	1	
C04L — T52L (T52L')	LODL		1					0	1	0	0	0	1	

PRINTED IN U.S. AMERICA 10-67

1031986

T REGISTER FLIP FLOP			SUB CLASS							CLASS					TYPE	
			12	11	10	9	8	7	6	5	4	3	2	1
C05L	T56L	BGEL	0					1	0	1	0	1	0	1
		BGAL	0				1		0	1	0	1	0	1
		BNEL	0		0	1			0	1	0	1	0	1
		BLEL	1					1	0	1	0	1	0	1
		BLAL	1				1		0	1	0	1	0	1
		BEQL	1		0	1			0	1	0	1	0	1
	EXCLD1	EXCL			1	0			0	1	0	1	0	1
		DUPL		1	0				0	1	0	1	0	1
	IFTL • EXCLD1	FCXL	0	0	1	1	0	0	0	1	0	1	0	1
		CCXL	1	0	1	1	0	0	0	1	0	1	0	1
		FFXL	0	1	1	1	0	0	0	1	0	1	0	1
		CFXL	1	1	1	1	0	0	0	1	0	1	0	1
C06L	TFBLD1	TFBL		1	0	0	0	0	0	1	1	0	0	1
		TIOL	1	1	0	1	0	0	0	1	1	0	0	1
		IPSL	0	1	0	1	0	0	0	1	1	0	0	1
	T57L	BFUL	1	0			1		0	1	1	0	0	1
		JFUL	1	1			1		0	1	1	0	0	1
		BBUL	1	0				1	0	1	1	0	0	1
		JBUL	1	1				1	0	1	1	0	0	1
		BFCL	0	0			1		0	1	1	0	0	1
		JFCL	0	1			1		0	1	1	0	0	1
		BBCL	0	0				1	0	1	1	0	0	1
		JBCL	0	1				1	0	1	1	0	0	1
	T54L	MSPL	1	0		1			0	1	1	0	0	1
		MSNL	0	0		1			0	1	1	0	0	1
		CSSL	0		1				0	1	1	0	0	1
		SSFL	1	1	1				0	1	1	0	0	1
CAGT71		RJPL						1	0	1	1	1	0	1
	780L	RNSL	RNML			0		1	0	1	1	1	0	1
			RSPL			1		1	0	1	1	1	0	1
			REWL				1		0	1	1	1	0	1
	T66L	INDL	INDL		0		0	1	1	0	0	0	0	1
			LLLL		1		1	1	1	0	0	0	0	1
			FXSL		1		0	1	1	0	0	0	0	1

T REGISTER FLIP FLOP			SUB CLASS						CLASS					TYPE	
			12	11	10	9	8	7	6	5	4	3	2	1
T66L	MAVL	MDVL				0		1	1	0	0	0	0	1
		MDAL				1		1	1	0	0	0	0	1
	MSOL	MSOL	0				1	0	1	0	0	0	0	1
		ECML	1				1	0	1	0	0	0	0	1
		VFIL	x	x	x	x	x	x	1	0	0	1	0	1
		ZFNL	y	y	y	y	0	0	1	0	1	0	0	1
		ZBNL	x	x	x	x	0	1	1	0	1	0	0	1
		ZFDL	x	x	x	x	1	0	1	0	1	0	0	1
		ZBDL	x	x	x	x	1	1	1	0	1	0	0	1
		DELL	0	0	0	0	0	0	1	0	1	0	0	1
		DIAL	y	y	y	y	y	y	1	0	1	1	0	1
		NOOP	0	0	0	0	0	0	1	0	1	1	0	1
		DIBL	y	y	y	y	y	y	1	1	0	0	0	1
		VARL	0	0	0	0	0	0	1	1	0	0	0	1
T71L		TRFL	x	x	x	x	x	x	1	1	0	1	0	1
	T72L	CFLL	x	x	x	x	x	x	1	1	1	0	0	1
		CFEL	x	x	x	x	x	x	1	1	1	1	0	1
		LTSL	x	x	x	x	x	x	x	x	x	x	0	0
	ODCL	OCSL	x	x	x	x	x	x	x	x	x	x	1	0
		DCSL	x	x	x	x	x	x	x	x	x	x	1	1
	T7OL	REBL	x	x	x	x	x	x	1	1	0	1	0	1
		SEBL	x	x	x	x	x	x	1	1	0	1	0	0
T62L T27L	T21L	T23L	SNEL	x	x	x	x	x	1	1	0	0	0	1
			SEQL	x	x	x	x	x	1	1	0	0	0	0
			SGTL	x	x	x	x	x	1	1	0	0	1	1
			SGEL	x	x	x	x	x	1	1	0	0	1	0
		FASL	FADL	x	x	x	x	x	1	1	1	0	1	1
			FSUL	x	x	x	x	x	1	1	1	0	1	0
			SLTL	x	x	x	x	x	1	1	1	0	0	1
			SLEL	x	x	x	x	x	1	1	1	0	0	0
			TSDL	x	x	x	x	x	1	1	1	1	1	1
			TZDL	x	x	x	x	x	1	1	1	1	1	0
			TNDL	x	x	x	x	x	1	1	1	1	0	1
			TPDL	x	x	x	x	x	1	1	1	1	0	0

PRINTED IN U.S. AMERICA 10-67

1031986

T REGISTER FLIP FLOP		SUB CLASS 12	11	10	9	8	7	CLASS 6	5	4	3	TYPE 2	1
	ICOL	x	x	x	x	x	x	1	1	0	1	1	1
	OCOL	x	x	x	x	x	x	1	1	0	1	1	0
	SSPL	x	x	x	x	x	x	0	1	0	0	1	0
T40L	FAXL	x	x	x	x	x	x	0	1	1	0	1	1
T40L	FSXL	x	x	x	x	x	x	0	1	1	0	1	0
T26L	FSSL	x	x	x	x	x	x	0	1	1	0	0	1
T26L	RSSL	x	x	x	x	x	x	0	1	1	0	0	0
T22L T34L	TGTL	x	x	x	x	x	x	0	1	0	1	1	1
T22L T34L	TGEL	x	x	x	x	x	x	0	1	0	1	1	0
T22L T34L	TNEL	x	x	x	x	x	x	0	1	0	1	0	1
T22L T34L	TEQL	x	x	x	x	x	x	0	1	0	1	0	0
T22L T34L	TLTL	x	x	x	x	x	x	0	1	1	1	0	1
T22L T34L	TLEL	x	x	x	x	x	x	0	1	1	1	0	0
T22L T34L	TANL	x	x	x	x	x	x	0	1	1	1	1	0
T22L T34L	TEBL	x	x	x	x	x	x	0	1	1	1	1	1
NOT ASSIGNED								0	1	0	0	1	1
NOT ASSIGNED								0	1	0	0	0	1
NOT ASSIGNED								0	1	0	0	0	0
T30L T35L T29L	CJOL	x	x	x	x	x	x	1	0	0	1	0	0
T30L T35L T29L	JOLL	x	x	x	x	x	x	1	0	0	1	1	0
T30L T35L T29L	CFJL	x	x	x	x	x	x	1	0	0	1	0	1
T30L T35L T29L	FWJL	x	x	x	x	x	x	1	0	0	1	1	1
T30L T35L	CRJL	x	x	x	x	x	x	1	0	1	1	0	1
T30L T35L	REJL	x	x	x	x	x	x	1	0	1	1	1	1
	CLRL	x	x	x	x	x	x	1	0	0	0	1	1
	SETL	x	x	x	x	x	x	1	0	0	0	1	0
	STAL	x	x	x	x	x	x	1	0	0	0	0	1
	INTL	x	x	x	x	x	x	1	0	0	0	0	0
	SSAL	x	x	x	x	x	x	1	0	1	1	1	0
T24L	RSAL	x	x	x	x	x	x	1	0	1	0	1	1
T24L	ENLL	x	x	x	x	x	x	1	0	1	0	0	1
T24L	BELL	x	x	x	x	x	x	1	0	1	0	1	0
T24L T37L	RPAL	x	x	x	x	x	x	1	0	1	0	0	0
T24L T37L	STPL	x	x	x	x	x	x	1	0	1	1	0	0
T25L	STSL	x	x	x	x	x	x	0	0	1	1	0	1
T25L T62L•T36L •T31L	STDL	x	x	x	x	x	x	0	0	1	1	0	0

T33L

T32L

T31L

T REGISTER FLIP FLOP		SUB CLASS 12	11	10	9	8	7	CLASS 6	5	4	3	TYPE 2	1
T62L • T31L	RSDL	x	x	x	x	x	x	0	0	1	1	1	1
T62L • T31L	FSDL	x	x	x	x	x	x	0	0	1	1	1	0
T28L T36L	SDPL	x	x	x	x	x	x	0	0	0	1	1	0
T28L	SDAL	x	x	x	x	x	x	0	0	0	1	1	1
T28L	TWDL	x	x	x	x	x	x	0	0	0	1	0	1
T28L T36L	RDAL	x	x	x	x	x	x	0	0	0	1	0	0
T28L	RECL	x	x	x	x	x	x	0	0	0	0	0	0
T28L	SBSL	x	x	x	x	x	x	0	0	0	0	1	1
T28L T20L	SBDL	x	x	x	x	x	x	0	0	0	0	1	0
NOT ASSIGNED								0	0	0	0	0	1
NOT ASSIGNED								0	0	1	0	1	1
T27L • T62L	TBZL	x	x	x	x	x	x	0	0	1	0	1	0
NOT ASSIGNED								0	0	1	0	0	0
	CHPL	x	x	0	x	x	0	0	0	1	0	0	1
	SFIL	x	x	1	x	x	0	0	0	1	0	0	1
INILD1	INIL	x	x	x	x	x	1	0	0	1	0	0	1
INILD1	IFTL	x		1			1	0	0	1	0	0	1
	SFTL		x	1	1		0	0	0	1	0	0	1

INILD1

NOTE

$y \equiv \neq 0$

x ≡ Don't Care

PRINTED IN U.S. AMERICA 10-67

1031986

I/O CONTROL/TAPE TIMING

NAME	SYNC POINT	OUTPUT POINT	TIMING	TAPE OPERATION/OPERATIONS
BF1M	ABC2U4	ABB4U0	96μs.	Continuous Backward Read (Hi Density)
BF2M	ABC2U4	ABB4F0	250μs.	Continuous Backward Read (Lo Density)
BTDM	AAD8P2	ADB0F0	6ms.	Continuous Write
BWIM	AAB6U0	AAB6U0	1.4ms.	Continuous Backward Read
DS1M	AAA4W5	AAB7U0	6μs.	Continuous Write (Hi Density)
DS2M	AAA4W5	AAB7F0	17μs.	Continuous Write (Lo Density)
LP1M	AAB7E9	AAB8U0	15ms.	Rewind Followed by a Read
LP1M	ABC2U4	AAB8F0	300μs.	Continuous Write (Hi Density)
LP2M	ABC2U4	AAB9F0	850μs.	Continuous Write (Lo Density)
WGBM	AAD7P2	AAB4F0	67ms.	Rewind Followed by a Write
WGNM	AAD7P2	AAB4U0	4.4ms.	Continuous Write

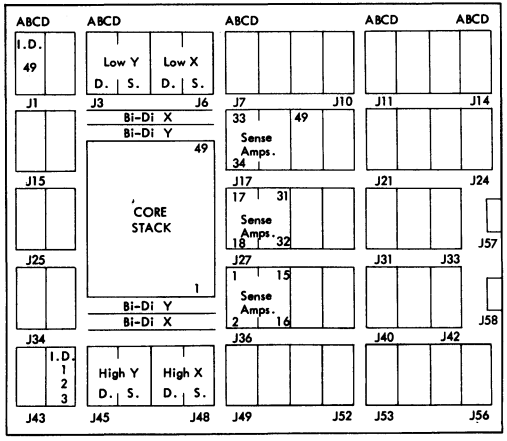
The following timings are for the tape delay circuits:

IM1M - 35μs. IM2M - 85μs. BRIM - 6.6ms.

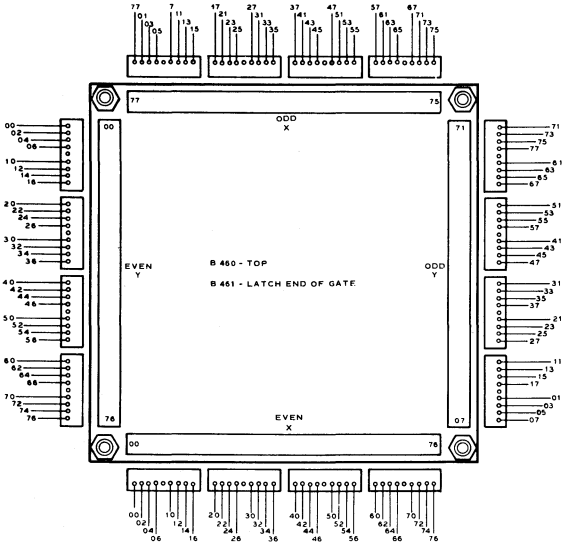
DSI = 1.7ms. MRD = 5.9ms.

TR 5903 is a one card I/O Descriptor Routine which can be used to facilitate performing the tape operations necessary to make the above adjustments.

B 460 CORE MEMORY CARD RACK - WIRING SIDE



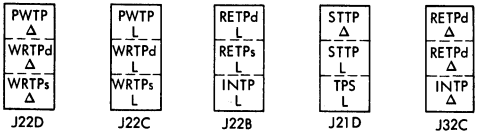
ADDRESS LINES



NOTE: B 460 - ODD - Y IS WIRING SIDE
B 461 - ODD - Y IS PACKAGE SIDE

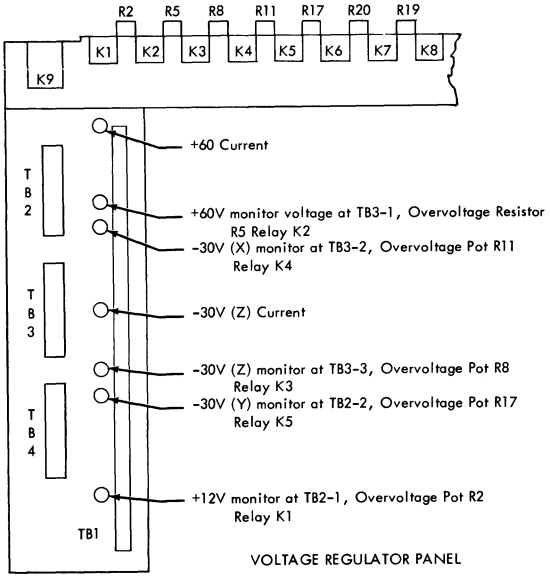
B 460 CORE MEMORY PULSE SHAPER TIMING TEST POINTS

Timing Pulse	Check Pulse at Location
RETPs	J31D30
RETPd	J31D20
INTP	J31D21
WRTPs	J31D6
WRTPd	J31D1
PWTP	J31D5
STTP	J30D1, 21 or 30



NOTE: Δ = DELAY ADJUSTMENT
L = PULSE LENGTH

B 460 CORE MEMORY OVER/UNDER VOLTAGE PANEL



VOLTAGE REGULATOR PANEL

B 461 CORE MEMORY TEST POINTS

MULTI TIMING TRAIN

	MS1M	ADC0X6
*	MR2M	ADC1X1
*	STTP	ABD2B5
	MRWP	ADC2U5
	MIHM	ACC3V9
*	MW2M	ADC4X1
	MICM	ABD2K4

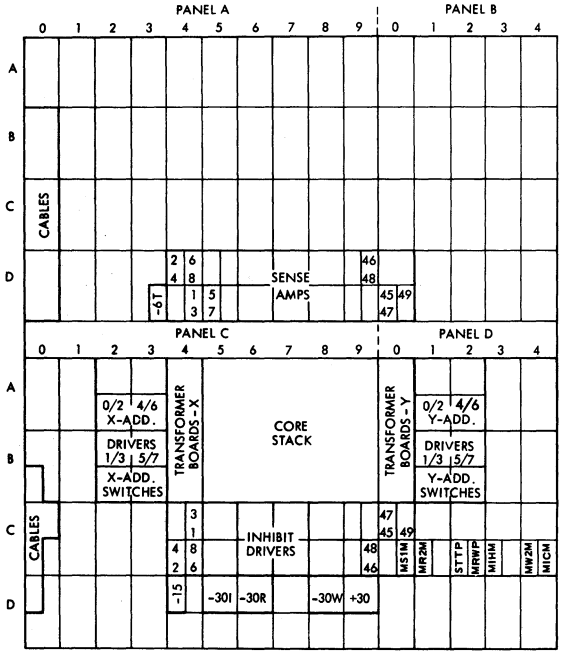
* Indicates those multis which are adjustable.

CLOCK PULSE - AAC7B7

Sense Amp Output	out - 1	out - 2
Even S.A.	J-0 or 7	J-2 or 9
ODD-S.A.	W-0 or 7	W-2 or 9

See Logical Rack wiring side for S.A. Locations!

B 461 LOGICAL GATE - WIRING SIDE



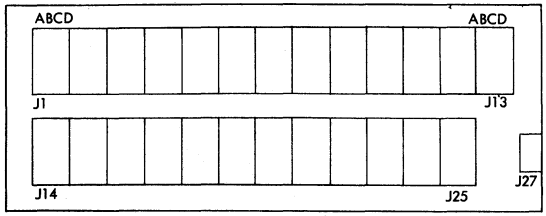
B 430 DRUM TEST POINTS

CMPD	-	J17A14	DRA-B	-	J7D5
CMD	-	J4D5	DRA-A	-	J7D1
DTP	-	J4D14	DRA-8	-	J7D6
DCLP	-	J4D22	DRA-4	-	J7D20
CMP	-	J18A22	DRA-2	-	J7D30
DMPD	-	J16C14	DRA-1	-	J7D21
WMPD	-	J4C20			

C.S.D. 3 circuit per card outputs

CKT-1	Pin 1
CKT-2	Pin 22
CKT-3	Pin 32

DRUM CARD RACK - WIRING SIDE



SECTION 10
TEST ROUTINES

TEST ROUTINES

Loader	5900	1109 4588
Chaining	5901	1109 4596
Card Lister	5902	1109 4562
N/O Utility	5903	1112 4088
I/O Test	5340	1106 2577
SPO and Keybd.	5556	1109 4554
Interaction	5558	1109 4570
Drum	5222	1109 4604 (Optional)
P.T.R.	5559	1109 4380 (Optional)
P.T.P.	5560	1112 4070 (Optional)
Mag. Tape MTR		1121 5027
Mag. Tape MARG		1119 9213
Proc. MTR Tape		1118 7614
Proc. MTR Loader		1118 7507
Proc. MTR Card Deck		1118 7531
Core MTR	1000	1118 2151
Core MTR	1100	1118 2185
Disk TR	Part 1	1117 0677
	Part 2	1117 0701
	Part 3	1117 0735
Data Comm TW/TWX/ACU		1126 7961 (Optional)
H120 Adapter		1142 1708 (Optional)
1050 Adapter		1142 1534 (Optional)
1004 Adapter		1142 1617 (Optional)

PRINTED IN U.S. AMERICA 10-67

1031986

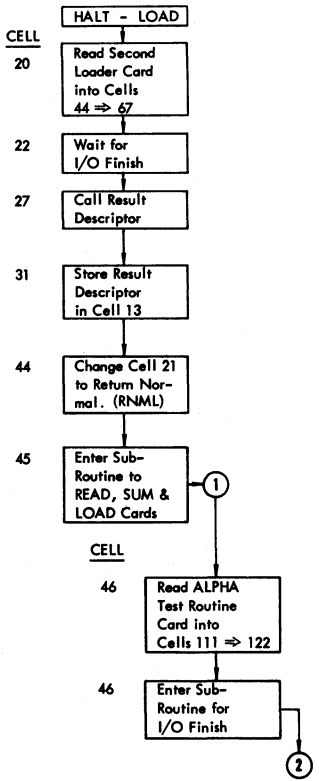
TEST ROUTINE LOADER - TR5900

LOADER CARD DESCRIPTION

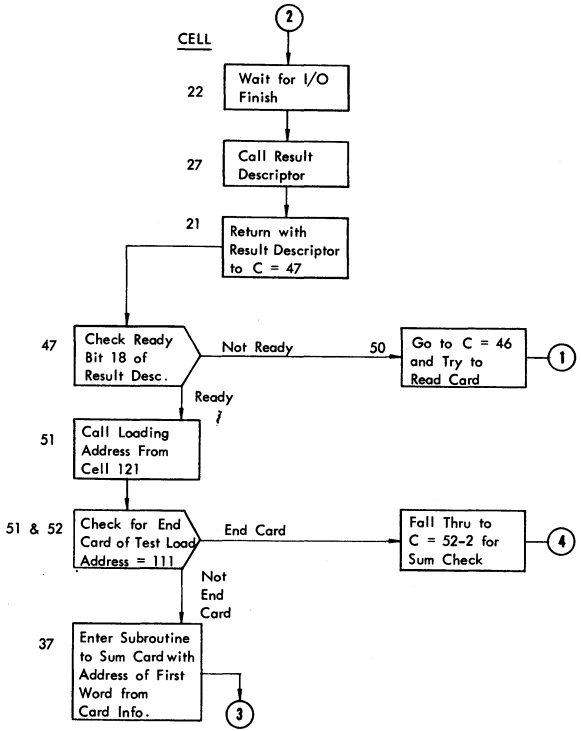
1. Two Card Binary Loader.
2. Card Identification in Column 80.
3. Reads two Card Loader into Cells 20 ⇒ 67.
4. Stores Result Descriptor of Binary Card Read in Cell 13.
5. Reads ALPHA Test Routine Cards into Input Buffer Cells 111 ⇒ 122.
6. Uses Loading Address in Columns 65 ⇒ 67 to transfer the first 8 (Octal 10) words from card into Memory.
7. Loads Cards into Memory until the End card of routine is read.
8. If Card Reader goes not ready, retry card, read until reader is made ready.
9. Sums all words from Test Routine Cards (24 High Order Bit added to 24 Low Order bits of each word) except end card and stores card sum in Cell 26.
10. Compare Card Sum with Sum Total from End Card (Cell 117).
11. On Sum Error Prints (SPO) SUM ERR, Stores Card Sum in Cell 117, Calls SPO Result Descriptor and Card Sum and then dynamically halts at C = 56.
12. At End of Read, Sum and Load Routine branches to C = 111. First syllable from end card to start Test Routine Read.

TEST ROUTINE LOADER - TR 5900 (Continued)

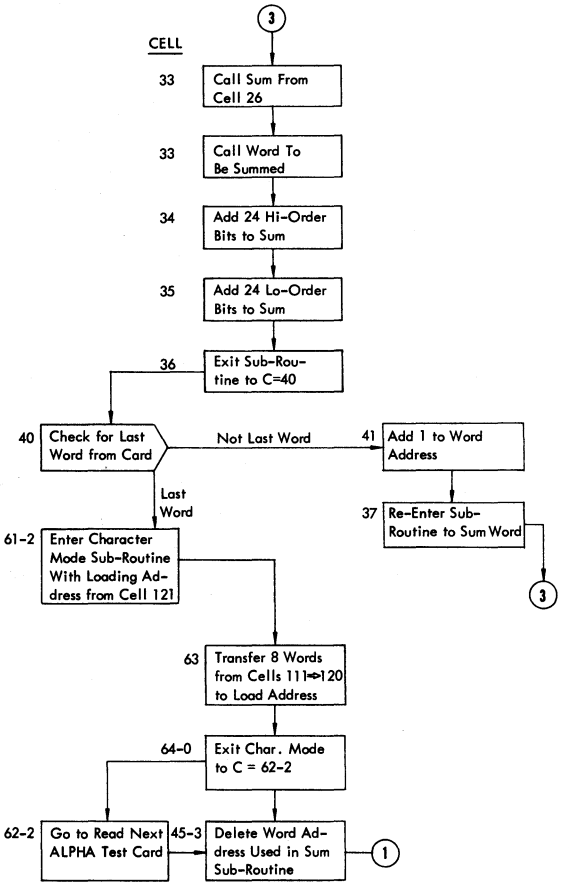
FLOW CHARTS



TEST ROUTINE LOADER - TR 5900 (Continued)



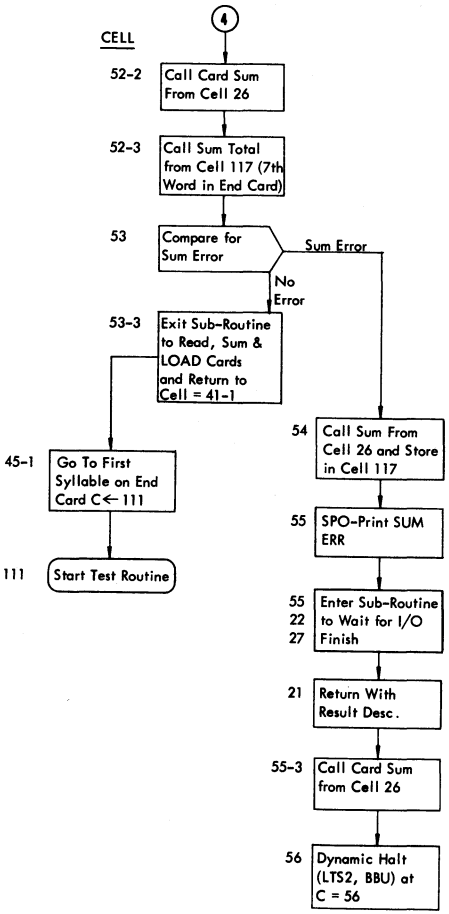
TEST ROUTINE LOADER - TR 5900 (Continued)



PRINTED IN U.S. AMERICA 10-67

1031986

TEST ROUTINE LOADER - TR 5900 (Continued)



FIRST CARD TEST ROUTINE LOADER - TR 5900

20-0	0114	LTS-23	Read Second Binary Loader Card into Cells 44 ⇒ 67
1	4411	I00L	
2	0020	LTS-4	
3	4231	BFU	Go to C = 22
21-0	0054	LTS-13	NOTE: This syllable to be Changed to 0235 RNWL
1	0421	BSD	
2	0117	DCS23	Store Binary Result Desc. in Cell 13
3	4231	BFU	To C = 44
22-0	0211	IINC	
1	0014	LTS3	Wait For I/O Finish
2	4131	BBU	
3	0000		
23-0	5240		
1	0004		Binary Card Read Descriptor to Cells 44 ⇒ 67
2	4000		
3	0044		
24-0	7500		
1	0000		Word Mode Prog. Desc. to C = 22
2	0000		Sub-Routine - Wait for I/O Finish
3	0022		
25-0	7700		
1	0000		
2	0000		Character Mode Program Descriptor
3	0063		To Transfer Words at C = 63
26-0	0000		
1	0000		Card Sum Storage
2	0000		Initially = Zero
3	0000		
27-0	0062	OCS-14	Call Result Descriptor I/O 1 Finish Routine
1	0055	No Op	CHPL Option
2	0160	LTS34	
3	4131	BBU	Go to C = 21 (RNWL)
30-0	0066	OCS-15	
1	0055	No Op	I/O 2 Finish Routine
2	0200	LTS40	
3	4131	BBU	
31-0	0072	OCS-16	
1	0055	No Op	I/O 3 Finish Routine
2	0220	LTS-44	
3	4131	BBU	
32-0	0076	OCS-17	
1	0055	No Op	I/O 4 Finish Routine
2	0240	LTS50	
3	4131	BBU	
33-0	0132	OCS-26	Sub-Routine to Sum Word
1	0000	LTS-0	Call Sum
2	7012	OCS F-2	Call Word Address
3	2021	LODL	Load Word from Card (F-2 Address)

PRINTED IN U.S. AMERICA 10-67 1031986

FIRST CARD TEST ROUTINE LOADER - TR 5900 (Continued)

```
34-0   4061   DIB-40
  1    3065   TRF  24 Bits   Transfer 24 Hi-Order bits to LTS-0
  2    0101   AD1L           Add to Sum
  3    0000   LTS-0

35-0   7012   OCS F-2
  1    2021   LODL           Reload Word from Card
  2    4055   DIA 40
  3    3065   TRF  24 Bits   Transfer 24 Lo-Order bits to LTS-0

36-0   0101   AD1L           Add to Sum
  1    0130   LTS-26
  2    0421   BSD            Store Sum in Cell 26
  3    0435   REWL           Exit Word Mode to C = 40

37-0   0444   LTS-111   Address of First Word from Card
  1    2025   DUPL      F-2 Parameter
  2    0441   MSOL      Mark Stack and Enter Sub-Routine
  3    0333   DCS-66    at C = 33 to Sum Word

40-0   0510   LTS-122   122 = Last Word on Card
  1    4225   B   A     If Word Address (B) is Less Than 122 (A) Then
  2    0410   LTS-102   Continue (C = 41 - 0)
  3    0231   BFC       Else Go To C = 61-2

41-0   0004   LTS-1
  1    0101   AD1L      Add 1 to Word Address
  2    0054
  3    4131              Go to DUPL at C = 37-1

42-0   5240
  1    0000              Alpha Card Read Descriptor
  2    4000              To Cells 111 ⇒ 122
  3    0111

43-0   0000
  1    0000              Card Identification  First Loader
  2    0000              Column 80 = All Bits Except 1 Punched
  3    7377
```

SECOND CARD TEST ROUTINE LOADER

```
44-0   0336   OCS-67    Store 0235 RNML from Cell 67 into Cell 22
  1    0104   LTS-21    (Return Normal Operator for I/O Finish
  2    0421   BSD       Routine)
  3    0441   MSOL      Mark Stack and Enter Sub-Routine at C = 46

45-0   0302   OCS-60    To Read, Sum and Load Cards
  1    0213   DCS-42    End Load, Go To Test Routine C = 111
  2    4231   BFU
  3    0065   DFL       Delete Word Address used in Sub-Routine

46-0   0210   LTS-42    Sub-Routine to Read, Sum and Load
  1    4411   100L          Read Alpha Test Routine Card
  2    0441   MSOL          Mark Stack and Sub-routine at
  3    0122   OCS-24        C = 22 to wait for I/O Finish and
```


SECOND CARD TEST ROUTINE LOADER (Continued)

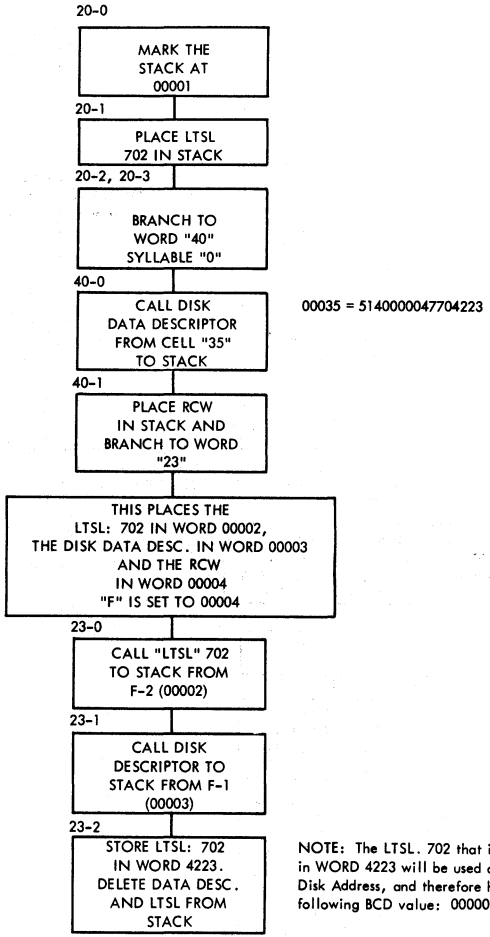
```
47-0   0000   LTS-0          Return with Result Descriptor
  1    5061   DIB-50 (Bit 18)
  2    0175   CFE  1 Bit     If Card Reader (Bit 18 Off)
  3    1025   EXCL

50-0   0065   DEL            (Delete Excess Word)
  1    0054   LTS-13         Then Continue (C = 50-3)
  2    0131   BBC            Else Go To C = 46 (Read Again)
  3    0213   DCS-42         Call First Word Address (111)

51-0   0506   OCS-121        Call Loading Address (Cell 121)
  1    2275   CFE  (18 bits) If End Card (Load Address = 111)
  2    1025   EXCL
  3    0065   DEL            (Delete Excess Word)

52-0   0270   LTS-56         Then Continue (C = 52-2)
  1    0131   BBC            Else Go To C = 37
  2    0132   OCS-26         Call Card Sum from 26
  3    0476   OCS-117        Call Sum Total from 117

53-0   4425   B = A          If Card Sum (B) Equals Total (A)
  1    0004   LTS-1          Then Continue (To C = 53-3)
  2    0231   BFC            Else Go To C = 54
  3    0435   REWL           Exit Sub-Routine to Read, Sum, Load

54-0   0132   OCS-26         Return to C = 45-1
  1    0474   LTS-117   Call and Store Errored Sum in Cell 117
  2    0421   BSD
  3    0324   LTS-65    Print Sum Error on SPO from Cell 57

55-0   4411   100L
  1    0441   MSOL      Mark Stack and Enter Sub-Routine to wait
  2    0122   OCS-24    for I/O Finish Return with Result Desc.
  3    0132   OCS-26    Call Errored Sum

56-0   0055   No Op          CHPL and Branch Option
  1    0055   No Op
  2    0010   LTS-2
  3    4131   BBU            Dynamic Halt

57-0   6264
  1    4460              Alpha for SUM ERR
  2    2551
  3    5137

60-0   7500
  1    0000              Word Mode Program Descriptor
  2    0000              To Read, Sum and Load Card at C = 46
  3    0046

61-0   0770   LTS-176   On Invalid Address Interrupt
  1    4131   BBU       Go To C = 22 and wait for I/O Finish
  2    0441   MSOL      Mark Stack
  3    0444   LTS-111   F-2 Parameter Address of First Word
```

PRINTED IN U.S. AMERICA 10-67 1031986

SECOND CARD TEST ROUTINE LOADER (Continued)

```
62-0   0506   OCS-121   F-1 Parameter Loading Address
  1    0127   DSC-25    Enter Character Mode at C = 63
  2    0324   LTS-65
  3    4131   BBU       Go To C = 45-3

63-0   0253   RSA F-2      SI ← (F-2);      SI ← 111
  1    0106   SOP F-1      DI ← Loc (F-1);  DI ← F-1
  2    0007   SDA          DI ← DC;         DI ← Load Address
  3    1005   TWD 10       DS ← 8 Wds;      Transfer 10 Wds.

64-0   0000   RECL         End              Exit Character Mode
  1    0000
  2    0000
  3    0000

65-0   5740
  1    0000   SPO I/O Descriptor to Type Sum Error
  2    0000   From Cell 57
  3    0057

66-0   7500
  1    0000   Word Mode Programing Descriptor
  2    0000   To Sum Word at C = 33
  3    0033

67-0   0235   RNML      RNML To Be Stored in Cell 22 (Ref: C = 44)
  1    0000
  2    0000   Card Identification  Second Loader
  3    7577   Column 80  All Bits Except 2 Punched
```

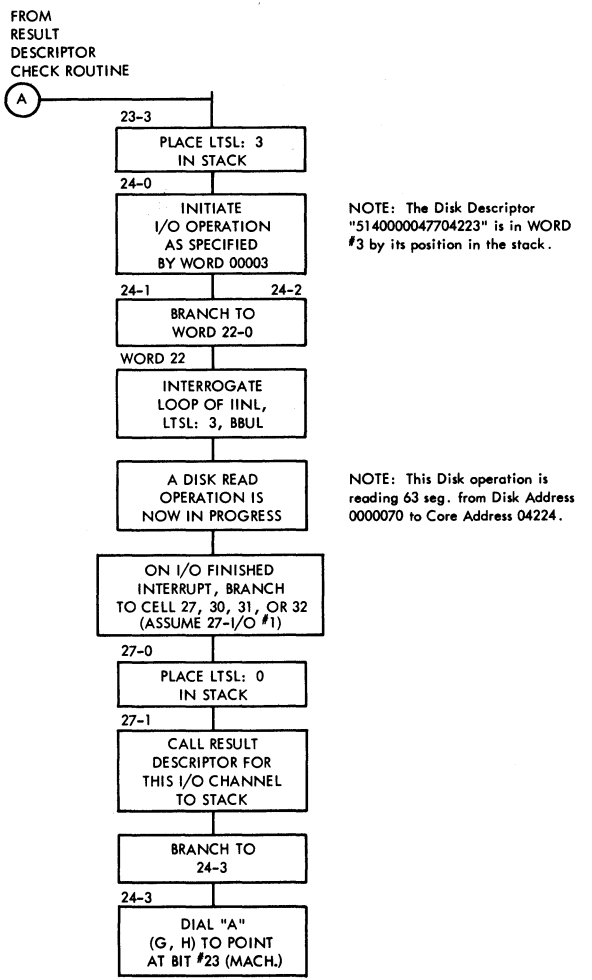

H/L FLOW CHART



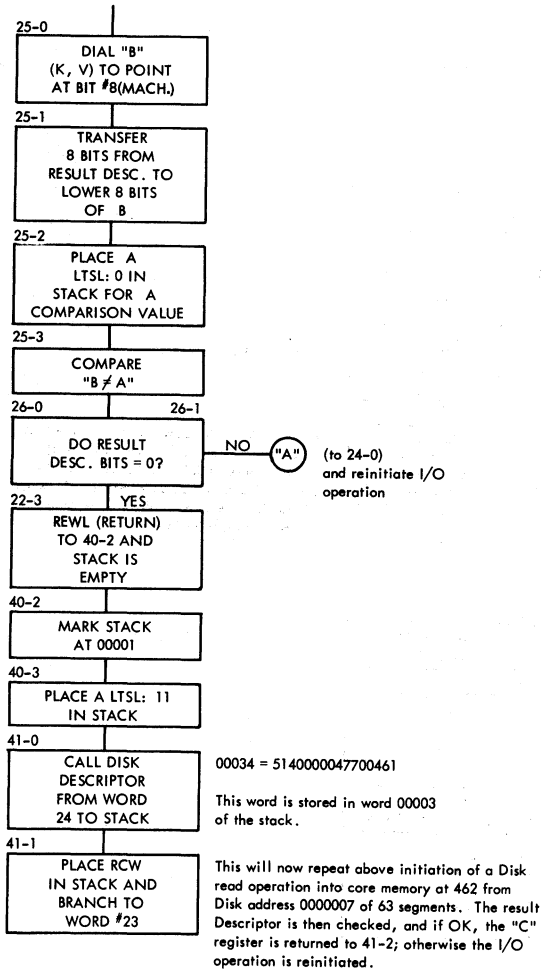
PRINTED IN U.S. AMERICA REVISED 3/70

1031986

H/L FLOW CHART (continued)



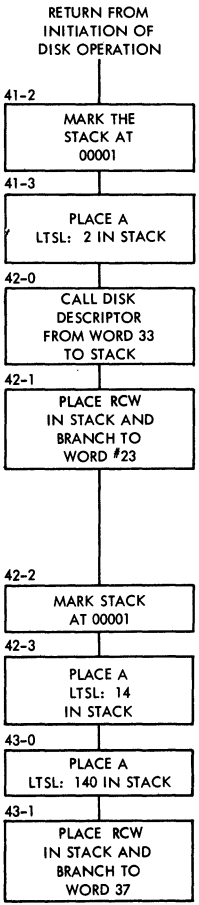
H/L FLOW CHART (continued)



PRINTED IN U.S. AMERICA REVISED 8/70

1031986

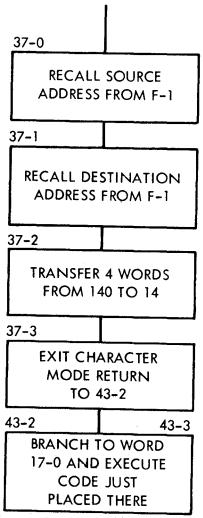
H/L FLOW CHART (continued)



00033 = 5140 0000 4070 0137

This now repeats the above operation of a Disk read operation into core memory address 0140 from Disk address 0000000, 7 segments. The result Descriptor is then checked; and if OK the "C" register is returned to 42-2, otherwise the I/O operation is reinitiated.

H/L FLOW CHART (continued)



This sets M to the address of 140

This sets "S" to the address of 14.

This will transfer the first four words of code brought initially from seg. #0 of Disk to words 14, 15, 16, 17.

PRINTED IN U.S. AMERICA REVISED 8/70 1031986

H/L CARD

20	0441				Mark Stack
		3410			Literal 702
			0360		Literal 74
				4231	Branch Forward Unconditional (40-0)
21	7500				Word Mode
		0000			Program
			0000		Descriptor
				0023	
22	0211				Interrogate Interrupt
		0014			Literal 3
			4131		Branch Backward Unconditional (22-0)
				0435	Exit
23	7012				Operand Call F-2
		7007			Descriptor Call F-1
			0421		B Store Destructive
				0014	Literal 3
24	4411				Initiate I/O
		0054			Literal 13
			4131		Branch Backward Unconditional (22-0)
				4155	Dial A 41
25	6461				Dial B 64
		1065			Transfer Bits 10
			0000		Literal 0
				0425	B Not Equal to A
26	0074				Literal 17
		0131			Branch Backward Conditional (22-3)
			0064		Literal 15
				4131	Branch Backward Unconditional (23-3)
27	0000				Literal 0
		0062			Operand Call 14
			0064		Literal 15
				4131	Branch Backward Unconditional (24-3)
30	0000				Literal 0
		0066			Operand Call 15
			0104		Literal 21
				4131	Branch Backward Unconditional (24-3)

H/L CARD (continued)

31	0000				Literal 0
		0072			Operand Call 16
			0124		Literal 25
				4131	Branch Backward Unconditional (24-3)
32	0000				Literal 0
		0076			Operand Call 17
			0144		Literal 31
				4131	Branch Backward Unconditional (24-3)
33	5140				Disk File Read Descriptor
		0000			7 segments
			4070		from address specified
				0137	in 0137
34	5140				Disk File Read Descriptor
		0000			$77_{(8)}$ Segments
			4770		from address specified
				0461	in 0461
35	5140				Disk File Read Descriptor
		0000			$77_{(8)}$ Segments
			4770		from address specified
				4223	in 4223
36	7700				Character Mode
		0000			Program
			0000		Descriptor
				0037	
37	0153				Recall Source Address F-1
		0204			Recall Destination Address F-2
			0405		Transfer Words 04
				0000	Exit Character Mode
40	0167				Descriptor Call 35
		0106			Operand Call 21
			0441		Mark Stack
				0044	Literal 11
41	0163				Descriptor Call 34
		0106			Operand Call 21
			0441		Mark Stack
				0010	Literal 2

PRINTED IN U.S. AMERICA REVISED 8/70

1031986

H/L CARD (continued)

42	0157				Descriptor Call 33
		0106			Operand Call 21
			0441		Mark Stack
				0060	Literal 14
43	0600				Literal 140
		0172			Operand Call 36
			0520		Literal 124
				4131	Branch Backward Unconditional (17-0)

Initialization Code Brought in by H/L Card

14	7700				Character Mode	
		0000			Program	
			0000		Descriptor	
				0015		
15	0253				Recall Source Address F-2	the 3969 words starting at 00160 are relocated starting at 00020
		0104			Recall Destination address F-1	
			7752		Begin Loop 63_{10}	
				7705	Transfer Words 63_{10}	
16	0051				End Loop	
		0000			Exit Character Mode	
			0000			
				0000		
17	0441				Mark Stack	
		0700			Literal 160_8	
			0100		Literal 20_8	
				0062	Operand Call 14_8	

NOTE: Enter at 17-0 from branch command at 43-3 of the H/L Card.

Operating Conditions

1. Timer can be on.
2. Printer Finished or Keyboard Request will stop the program.
3. Will work on any I/O Channel.

ESPOL LOAD CARD

20	0104				Literal 21
		4411			INITIATE I/O
			0020		Literal 4
				4231	Branch Forward Unconditional (22-0)
21	5240				Card Read Descriptor
		1200			Alpha 12_8 Words
			4000		CRA
				0044	
22	4455				Dial A 44 (Bit 20)
		0211			Interrogate Interrupt
			0020		Literal 4
				4131	Branch Backward Unconditional (22-0)
23	7700				Character Mode
		0000			Program Descriptor
			0000		
				0024	
24	0453				Recall Source Address F-4
		0304			Recall Destination Address F-3
			0243		Call Repeat Field F-2
				0005	Transfer Words
25	0000				Exit Character Mode
		0065			Transfer bits 00
			0100		Literal 20
				4131	Branch Backward Unconditional (22-0)
26	0110				Literal 22
		4131			Branch Backward Unconditional (22-0)
			0055		Dial A 00
				0055	Dial A 00
27	0000				Literal 0
		0062			Operand Call 14
			0050		Literal 12
				4231	Branch Forward Unconditional (32-2)
30	0000				Literal 0
		0066			Operand Call 15
			0030		Literal 6
				4231	Branch Forward Unconditional (32-2)

PRINTED IN U.S. AMERICA 10-67

1031986

ESPOL LOAD CARD (continued)

31	0000				Literal 0
		0072			Operand Call 16
			0010		Literal 2
				4231	Branch Forward Unconditional (32-2)
32	0000				Literal 0
		0076			Operand Call 17
			7561		Dial B 75
				0165	Transfer Bits 01
33	0010				Literal 2
		0231			Branch Forward Conditional (34-0)
			0010		Literal 2
				4131	Branch Backward Unconditional (33-2)
34	0004				Literal 1
		0107			Descriptor Call 21
			2025		Duplicate
				0044	Literal 11
35	0106				Operand Call 21
		2025			Duplicate
			3355		Dial A 33
				4061	Dial B 40
36	2565				Transfer Bits 25
		2025			Duplicate
			2265		Transfer Bits 22
				2025	Duplicate
37	1765				Transfer Bits 17
		2025			Duplicate
			1465		Transfer Bits 14
				5355	Dial A 53
40	5361				Dial B 53
		1765			Transfer Bits 17
			0000		Literal 0
				0044	Literal 11
41	0106				Operand Call 21
		2025			Duplicate
			1555		Dial A 15
				2261	Dial B 22

ESPOL LOAD CARD (continued)

42	0165				Transfer Bits 01
		2255			Dial A 22
			7261		Dial B 72
				0465	Transfer Bits 04
43	0441				Mark Stack
		0116			Operand Call 23
			0500		Literal 120
				4131	Branch Backward Unconditional (20-0)

ESPOL TRANSFER CARD

11	7500				Word Mode
		0000			Program
			0000		Descriptor
				0012	
12	0004				Literal 1
		5355			Dial A 53 (C Field)
			3061		Dial B 30 (F Field)
				1765	Transfer Bits 15_{10}
13	7006				Operand Call F-1
		0004			Literal 1
			0421		B Store Destructive
				0435	Exit
14	7700				Character Mode
		0000			Program
			0000		Descriptor
				0015	
15	0253				Recall Source Address F-2
		0104			Recall Destination Address F-1
			7752		Begin Loop 63_{10}
				7705	Transfer Words 63_{10}
16	0051				End Loop
		0000			Exit Character Mode
			0441		Mark Stack
				0046	Operand Call 11

} the 3969 words starting at 00160 are relocated starting at 00020

PRINTED IN U.S. AMERICA 10-67 1031986

ESPOL TRANSFER CARD (continued)

17	0441				Mark Stack
		0700			Literal 160
			0100		Literal 20
				0062	Operand Call 14
20	0040				Literal 10
		4131			Branch Backward Unconditional (16-2)
			0000		
				0000	

NOTE: 20 is overlayed by character mode word transfer in 15-3.