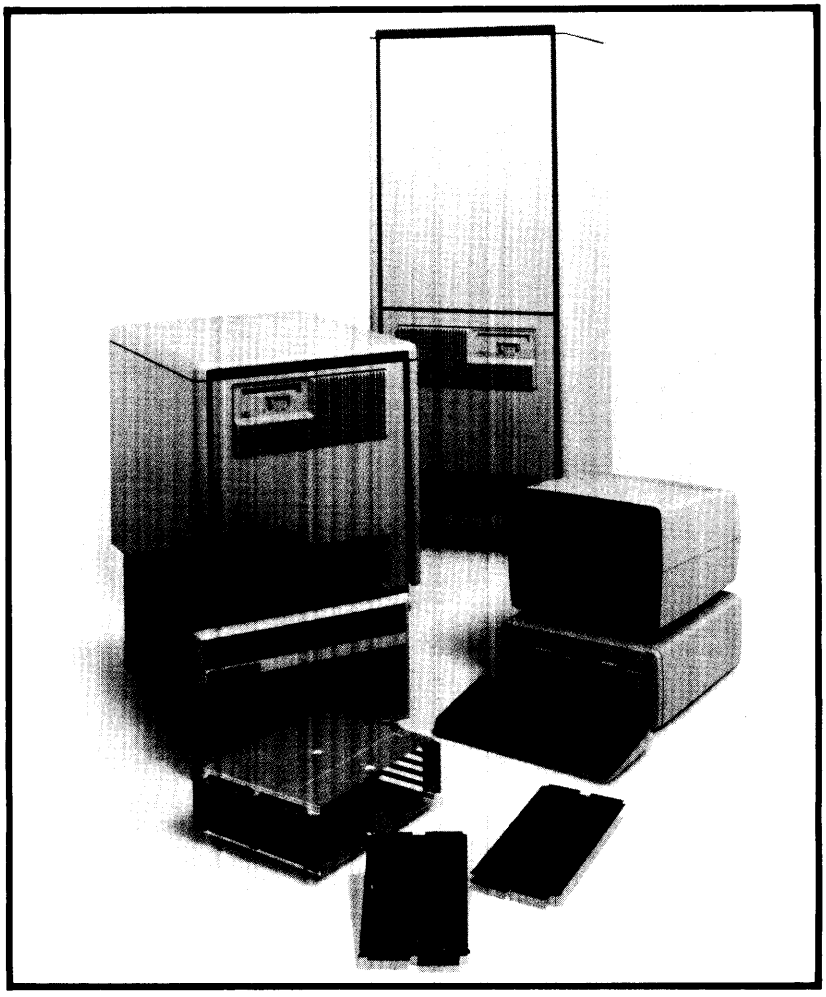
HEWLETT PACKARD

HP 1000 A600/A600+ Computer

Engineering and Reference Documentation — Vol. 2

HP 1000 A-Series



HP 1000 A600/A600+ Computer

Engineering and Reference Documentation

Volume 2

HEWLETT-PACKARD COMPANY
Data Systems Division
11000 Wolfe Road
Cupertino, California 95014

Vol. 2
MANUAL PART NO. 02156-90003
Printed in U.S.A. March 1983
U0684

Table of Contents

VOLUME 2:

Appendix A POWER SUPPLY

INTRODUCTION A-1
440-WATT SUPPLY A-1
LOGICAL SIGNALS A-2
PON+ A-2
PFW- A-3
MLOST- A-3
MECHANICAL SPECIFICATIONS (440W SUPPLY) A-3
ELECTRICAL CONNECTIONS (440W SUPPLY) A-5
ELECTRICAL SPECIFICATIONS (440W SUPPLY) A-8
ENVIRONMENTAL SPECIFICATIONS (440W SUPPLY) A-11
REPLACEABLE PARTS (440W SUPPLY) A-12
300-WATT SUPPLY A-27
LOGICAL SIGNALS A-28
MECHANICAL SPECIFICATIONS (300W SUPPLY) A-29
ELECTRICAL CONNECTIONS (300W SUPPLY) A-30
ELECTRICAL SPECIFICATIONS (300W SUPPLY) A-33
ENVIRONMENTAL SPECIFICATIONS (300W SUPPLY) A-36
REPLACEABLE PARTS (300W SUPPLY) A-36
HP 12154A BATTERY BACKUP MODULE A-43
CONFIGURATION A-43
PHYSICAL DESCRIPTION A-43
ELECTRICAL SPECIFICATION A-43
THEORY OF OPERATION A-43
REPLACEABLE PARTS A-44
HP 12159A 25 kHz POWER MODULE A-49
HP 12159A SPECIFICATIONS A-49
HP 12159A THEORY OF OPERATION A-51
REPLACEABLE PARTS A-51
25 kHz BACKPLANE POWER APPLICATIONS A-56
NON-ISOLATED POWER SUPPLY A-56
Purpose and Basic Design A-56
Preserving Purity of Input Sine Wave A-56
Rectifier Selection A-57
Input Noise Reduction A-58
Input Filtering A-58
Regulator Dissipation A-59
ISOLATED OR FLOATING DC POWER A-59
LOW-VOLTAGE, HIGH-CURRENT POWER SUPPLY A-60

Appendix B PROCESSOR CARD REFERENCE DATA

INSTRUCTION DECODE PAL (U1405) B-2
DESTINATION SPECIAL PAL (U507) B-3
A/B ADDRESS SPECIAL PAL (U802) B-4
INTERRUPT CONTROLLER NO. 1 PAL (U307) B-5
INTERRUPT CONTROLLER NO. 2 PAL (U407) B-6
ASG SKIP SPECIAL PAL (U807) B-7
INTERRUPT JUMP TABLE FPLA (U207) B-8
SOURCE SPECIAL FPLA (U607) B-10
I/O HANDSHAKE FPLA (U709) B-12

Appendix C VCP, LOADERS, AND SELF-TEST PROGRAMS

Appendix D A600/A600+ BASESET MICROCODE (A600 Code is Optional)

LIST OF ILLUSTRATIONS (Volume 2)

Figure A-1. 440-Watt Power Supply Block Diagram A-2
Figure A-2. Dimensions and Connector Locations (440W Supply) A-4
Figure A-3. 440 Watt Power Supply Connector Diagram A-5
Figure A-4. Parts Locations for Board 1, 440W Supply A-13
Figure A-5. Parts Locations for Board 2, 440W Supply A-17
Figure A-6. Parts Locations for Battery Backup Card (BB500) A-20
Figure A-7. Parts Locations for 25 kHz Card (SW100) A-24
Figure A-8. 300-Watt Power Supply Block Diagram A-28
Figure A-9. Dimensions and Connector Locations (300W Supply) A-30
Figure A-10. 300-Watt Power Supply Connector Diagram A-31
Figure A-11. Parts Locations for 300W Supply A-37
Figure A-12. HP 12154A Battery Backup Module Schematic A-45
Figure A-13. HP 12154A Replaceable Parts Location Diagram A-46
Figure A-14. HP 12159A 25 kHz Power Module Schematic A-53
Figure A-15. HP 12159A Replaceable Parts Locations A-54
Figure A-16. On-Interface Regulated Power Supply A-57
Figure A-17. Multiple, Isolated, On-Interface Power Supplies A-60
Figure A-18. On-Interface, High Current Switching Power Supply A-61

LIST OF TABLES (Volume 2)

Table A-1. Connector Specifications (440W Supply) A-3
Table A-2. Electrical Connections (440W Supply) A-6
Table A-3. Input Electrical Specifications (440W Supply) A-8
Table A-4. Battery Input Specification (440W Supply) A-9
Table A-5. Output Electrical Specifications (440W Supply) A-9
Table A-6. Environmental Specifications (440W Supply) A-11
Table A-7. 440W Supply Board 1 Replaceable Parts A-14
Table A-8. 440W Supply Board 2 Replaceable Parts A-18
Table A-9. Battery Backup Card Option Replaceable Parts A-21
Table A-10. 25 kHz Card Option Replaceable Parts A-25
Table A-11. Connector Specifications (300W Supply) A-29
Table A-12. Electrical Connections (300W Supply) A-32
Table A-13. Input Electrical Specifications (300W Supply) A-33
Table A-14. Output Electrical Specifications (300W Supply) A-34
Table A-15. 300W Supply Replaceable Parts A-38
Table A-16. HP 12154A Replaceable Parts A-47
Table A-17. HP 12159A 25-kHz Module Electrical Specifications A-50
Table A-18. HP 12159A Electrical Connector (P1) Pin Definitions . . . A-51
Table A-19. HP 12159A Replaceable Parts A-55

POWER SUPPLY	APPENDIX A

A.1 INTRODUCTION

There are two power supplies used with the A-Series Computers A 440-watt supply is for the 20-slot backplane and a 300-watt supply is for the 16-slot backplane. Both supplies are modules that plug into the back (circuit trace side) of the appropriate backplane. The A-Series supplies are considered non-repairable in the field and, in case of failure, the entire unit should be replaced with an exchange unit from Hewlett-Packard and the original unit returned for repair.

This section of the manual provides information required to evaluate the supply's performance. Included are an overall operating description, control signal descriptions, mechanical and electrical specifications. Located at the back of this section are parts location diagrams (assembly drawings), parts lists, and schematics.

This section is divided into several main parts. The paragraphs under subheading A.2 cover the 440-watt supply, Part No. 0950-1671; the paragraphs under subheading A.3 cover the Micro/1000 300-watt supply, Part No. 0950-1646, the paragraphs under subheading A.4 cover the HP 12154A battery backup module for Micro/1000 systems, the paragraphs under subheading A.5 cover the HP 12159A 25 kHz module for Micro/1000 systems, and under subheading A.6 applications of the 25 kHz power are discussed.

A.2 440-WATT SUPPLY

The 440-watt supply, Part No. 0950-1671, is used with the 20-slot backplane. The supply operates from either 115 Vac or 230 Vac. There are four fans for cooling the power supply and the computer. The fans plug into either connector P7 for 115 Vac operation or into P8 for 230 Vac operation.

The supply has four dc outputs at +5V, +5V memory backup (+5M), +12V, and -12V. It also provides 25 kHz ac power that is used as a power source for certain I/O cards. The +5M battery backup (BB) and the 25 kHz ac outputs are optional and are provided by separate cards that plug into the supply. If the battery backup is not installed, a jumper card must be placed in the BB connector of the supply. A block diagram of the 440-Watt supply is shown in Figure A-1.

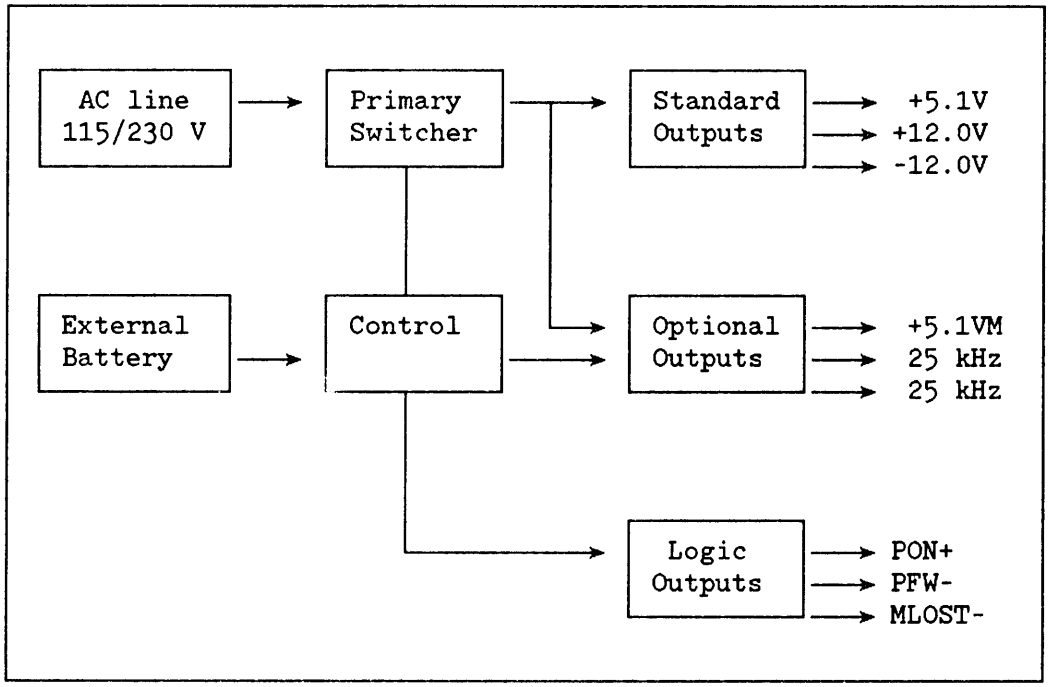


Figure A-1. 440-Watt Power Supply Block Diagram

A.2.1 LOGICAL SIGNALS

The power supply provides logical control signals to the computer to indicate power availability so that appropriate action can be taken.

A.2.1.1 PON+

PON+ is a signal that indicates the condition of the dc outputs. When the outputs are within specification, PON+ will be 2.4V to 5.2V. When the outputs are outside of specification, PON+ will be 0.2V plus or minus 0.2V. This definition includes the time when ac power is not applied (i.e., when ac power is down, the PON+ signal should be the out-of-specification condition.)

A.2.1.2 PFW-

The PFW- signal indicates the condition of ac power into the supply. When the input line voltage is above the "power fail trip point", PFW- is 2.4V to 5.2V. When the input line voltage is below the "power fail trip point", PFW- is 0.2V plus or minus 0.2V.

A.2.1.3 MLOST-

The MLOST- indicates the condition of the memory backup voltage as the main supply is being powered up. At all other times this signal is of no importance to the system. MLOST- is a pulse that is valid for 1 millisecond before and 5 millisecond after the rising edge of PON+. The MLOST- pulse during power up will be 2.4V to 5.2V if the memory supplies were within specification during the last power down. If the memory supplies are not within specification, MLOST- will be 0.2V plus or minus 0.2V.

A.2.2 MECHANICAL SPECIFICATIONS (440W SUPPLY)

The overall mechanical dimensions and connector locations of the 0950-1671 supply are shown in Figure A-2. The connector specifications are given in Table A-1.

The cooling air flow should be a minimum of 70 CFM of air flowing across the power board in the direction indicated in Figure A-2. Power supply assembly diagrams are provided at the rear of this section of the manual.

Table A-1. Connector Specifications (440W Supply)

CONNECTOR	AMP PART NO.	AMP MATING NO.
P1/P3	Edge Card	
P2/P4	Edge Card	
P5	9-350255-2	350240
P6	9-350264-2	350243
P7	207584-1	207396-1
P8	207584-1	207396-1
P9	207365-1	207360-1

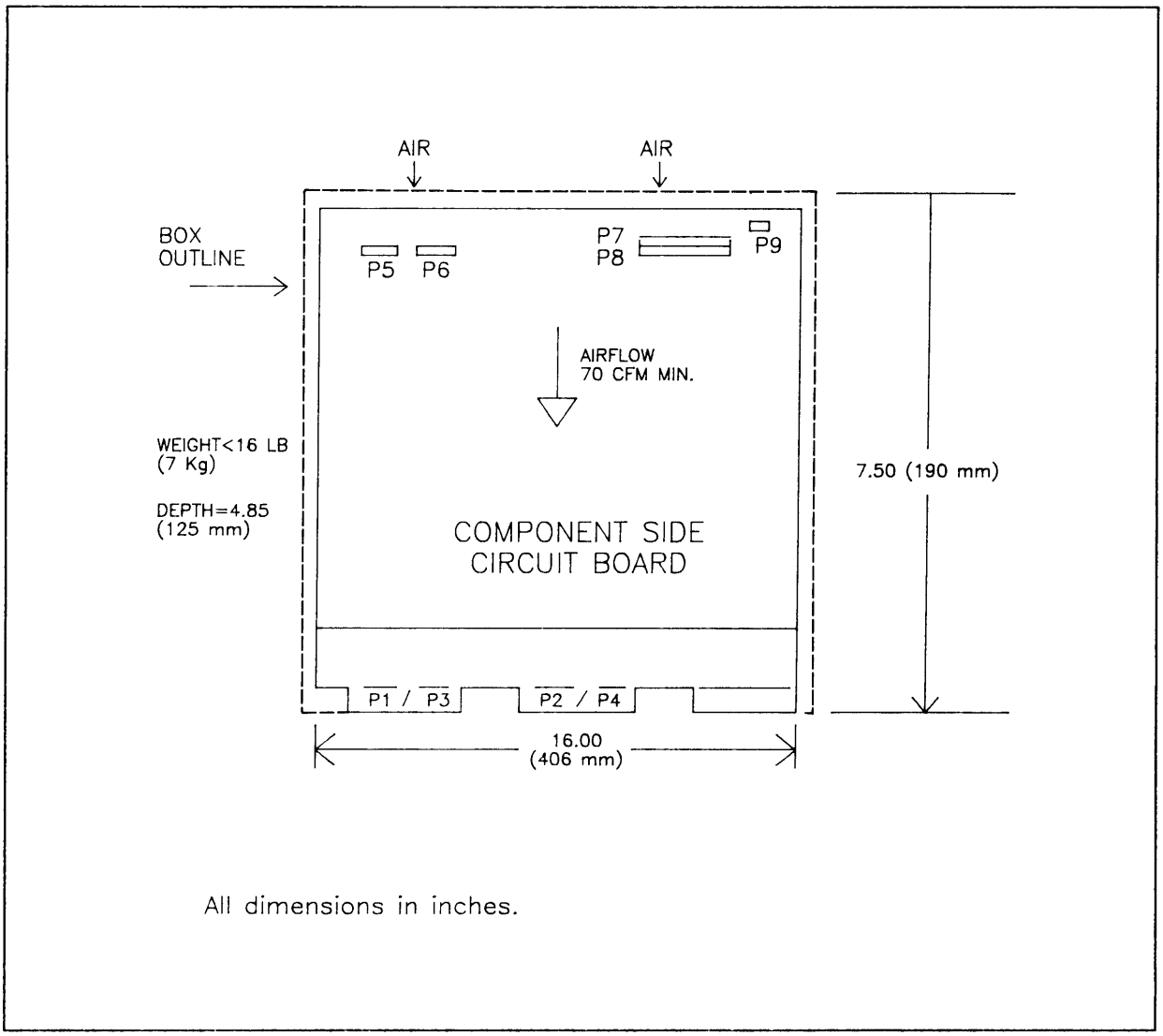


Figure A-2. Dimensions and Connector Locations (440W Supply)

A.2.3 ELECTRICAL CONNECTIONS (440W SUPPLY)

The electrical contacts for the 440 watt power supply are provided by nine connectors. Two of these are edge-card connectors that plug into the backplane. A power supply connector diagram is shown in Figure A-3, and the electrical connector pin definitions are given in Table A-2.

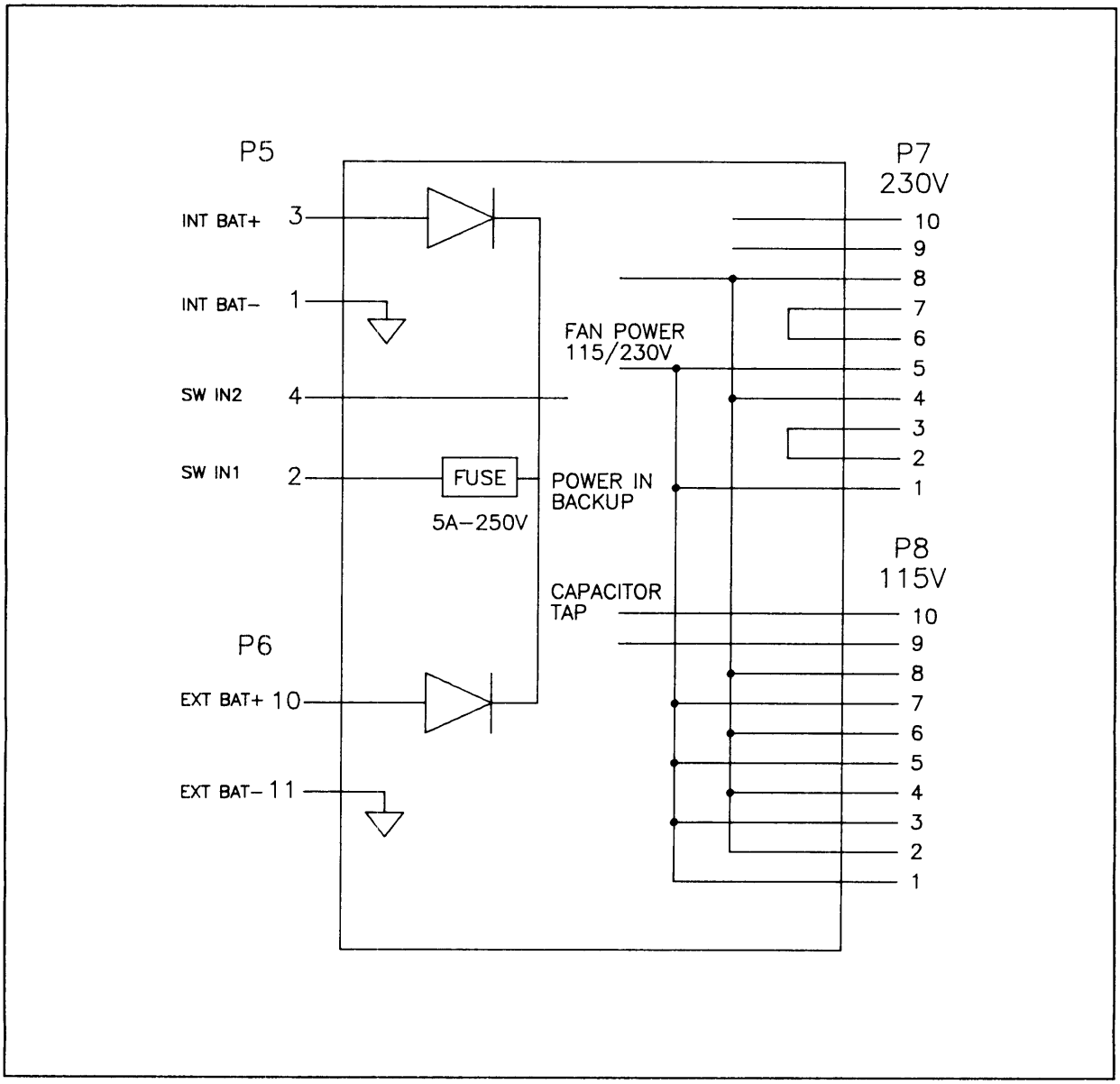


Figure A-3. 440 Watt Power Supply Connector Diagram

Table A-2. Electrical Connections (440W Supply)

P1 DC OUTPUT CONNECTOR (PC EDGE BOARD)	
Pin Number	**Signal Name**
1 thru 36	+5.1 Volts
37 thru 50	Common
P2 DC OUTPUT CONNECTOR (PC EDGE BOARD)	
Pin Number	**Signal Name**
1 thru 28	Common
29 thru 32	+12 Volts
33, 34	-12 Volts
35 thru 38	+5.1 Volts Memory Backup
39 thru 42	25 kHz Phase 1
43 thru 46	25 kHz Phase 2
47	PON+
48	PFW-
49	MLOST-
50	+5.1 Volts Memory Sense
P5 BATTERY SWITCH CONNECTOR	
Pin Number	**Signal Name**
2	Switch in 1
4	Switch in 2
1	Internal Battery -
3	Internal Battery +
P9 AC LINE INPUT	
Pin Number	**Signal Name**
1	AC Line
2	No Connection
3	AC Neutral

Table A-2. Electrical Connections (440W Supply) (Continued)

P7/P8 AC LINE CONFIGURATION / FANS	
Pin Number	**Signal Name**
1	Fan #1
2	Fan #1
3	Fan #2
4	Fan #2
5	Fan #3
6	Fan #3
7	Fan #4
8	Fan #4
9	230v / 115v
10	230v / 115v
P6 TEST POINTS / EXTERNAL BATTERY	
Pin Number	**Signal Name**
1	+5V Test
2	+12V Test
3	-12V Test
4	+5VM Test
5	PON
6	PFW
7	MLOST
8	25 kHz Test
9	25 kHz Test
10	Battery +
11	Battery -
12	Common

A.2.4 ELECTRICAL SPECIFICATIONS (440W SUPPLY)

The electrical specifications of the 440 watt supply are provided below in several tables. Ac line input specifications are given in Table A-3, battery input specifications are given in Table A-4, and supply output specifications are given in Table A-5.

Table A-3. Input Electrical Specifications (440W Supply)

AC LINE SPECIFICATIONS

These specifications do not include the power required for the fans.

	Min	Nominal	Max	
Range 1				
Voltage	84	120	140	Volts RMS
RNS Current (Max)	9.4	7.2	6.2	Amps
Inrush	-	-	136	Amps
Range 2				
Voltage	176	230	278	Volts RMS
RMS Current (Max)	4.7	3.7	3.1	Amps
Inrush	-	-	262	Amps
Carry Over	10.6	-	-	mSec
PFW Trip Point				
Range 1	-	-	84	Volts RMS
Range 2	-	-	176	Volts RMS
Line Frequency	47	60	67	Hz
Line Fuse	-	-	10	Amps
Input Power	-	-	700	Watts

Note: Power supply input operation permits input transients of up to 3000V for periods of not less than 10 s.

Table A-4. Battery Input Specification (440W Supply)

	MINIMUM	NOMINAL	MAXIMUM	
Battery Voltage	10.0	12.0	14.4	Volts
Discharge, Continuous	-	-	40.0	Amps
Internal Resistance	-	10.0	-	mohms

Note: 10.0V is the approximate input disconnect voltage. Disconnect occurs when Output #1 (5.1V) drops to 4.9V, as measured at the battery backup board (coincident with the assertion of MLOST-.

Table A-5. Output Electrical Specifications (440W Supply)

Maximum Dynamic Load: **10% over 10 microseconds**

Output Stress Conditions Allowed:

a. Supply will recover from a shorted regulated output and excessive ambient temperature.

b. Over rated operated temperature.

Output Regulation (Note 4):

Output # 1	Nominal Voltage	5.1	Volts	
	Maximum Current	70	Amps	(1) (5)
Regulation	0.0 to 3.0 Amps	+10%	-10%	
	3.0 to 6.2 Amps	+5%	-5%	
	6.2 to 70.0 Amps	+2%	-2%	
Output # 2	Nominal Voltage	12.0	Volts	
	Maximum Current	5.6	Amps	
Regulation	0.0 to .03 Amps	+10%	-10%	
	.03 to 5.6 Amps	+6%	-3%	
Output # 3	Nominal Voltage	-12.0	Volts	
	Maximum Current	3.5	Amps	
Regulation	0.0 to .10 Amps	+12%	-12%	
	.10 to 3.5 Amps	+6%	-6%	

Table A-5. Output Electrical Specifications (440W Supply) (Continued)

Output # 4 (opt.)	Nominal Voltage	5.1	Volts
	Maximum Current	10.0	Amps (2)
Regulation	0.0 to .10 Amps	+10%	-10%
	.10 to 10.0 Amps	+2%	-2%
Output # 5 (opt.)	Nominal Voltage	39	Volts RMS (5)
	Split Phase	19.5	Volts RMS
	Maximum Current	1.5	Amps
Regulation	0.0 to .02 Amps	+10%	-12%
	.02 to 1.5 Amps	+8%	-8%
Output # 6 (opt.)	Battery Charger		
	Minimum Current less than	.050	Amps (3)
	Maximum Current	.200	Amps
	Maximum Voltage	14.4	Volts
Output # 7	Fan Power		
	Nominal Voltage	115	Volts RMS
	Maximum Current	1.25	Amps

NOTES: (1) When no battery backup module is installed, the Output #4 current is supplied by Output #1. The total current drawn from Output #1 will not exceed 70 Amp.

(2) Output #4 shall be limited to 7 Amps when the 0950-1666 battery backup module is installed.

(3) When the battery is fully charged.

(4) Although the sum of the maximums listed above exceeds the 440 Watt specification of the power supply front end, not all of the outputs will be at maximum load at the same time and the actual maximum load will never exceed 440 Watts (not including fan power).

(5) When the maximum load is applied to Output #5 the load on Output #1 will not exceed 64 Amps.

A.2.5 ENVIRONMENTAL SPECIFICATIONS (440W SUPPLY)

The environmental specifications of the 0950-1671 440-watt power supply are provided in Table A-6.

Table A-6. Environmental Specifications (440W Supply)

Non Operating Temperature:
-40 deg C to 75 deg C

Operating Temperature:
0 deg C to 55 deg C
Type Tested
-5 deg C to 60 deg C (to insure margins)

Operating Survival Temperature:
-20 deg C to 65 deg C

Operating Humidity:
5% to 95% at 40 deg C wet bulb temperature

Vibration:

Sweep
From 5 to 55 Hz and back at a rate of one octave per minute, with an excursion of .015", for 15 minutes

Resonance
At each resonant point, not to exceed 4 points, dwell for 10 minutes at the following excursions:

5 - 10 Hz	.125"
11 - 25 Hz	.060"
25 - 55 Hz	.015"

Shock:

30g peak force applied as an 11 millisecond sine pulse. To be tested in each direction of each axis (6 tests).

Altitude:

Full operating temperature, at 440 Watts output power (not including fan power), at altitudes up to 10,000 feet, at 15,000 feet a derating of up to 10 deg C, of operating temperature, is allowed.

A.2.6 REPLACEABLE PARTS (440W SUPPLY)

Replaceable parts lists for the 440W power supply are provided in Tables A-7 and A-8. Table A-9 covers the battery backup board, and Table A-10 covers the 25 kHz sinewave card. The parts can be located in Figures A-4 through A-7. These tables and figures are located at the back of this section in front of the schematics.

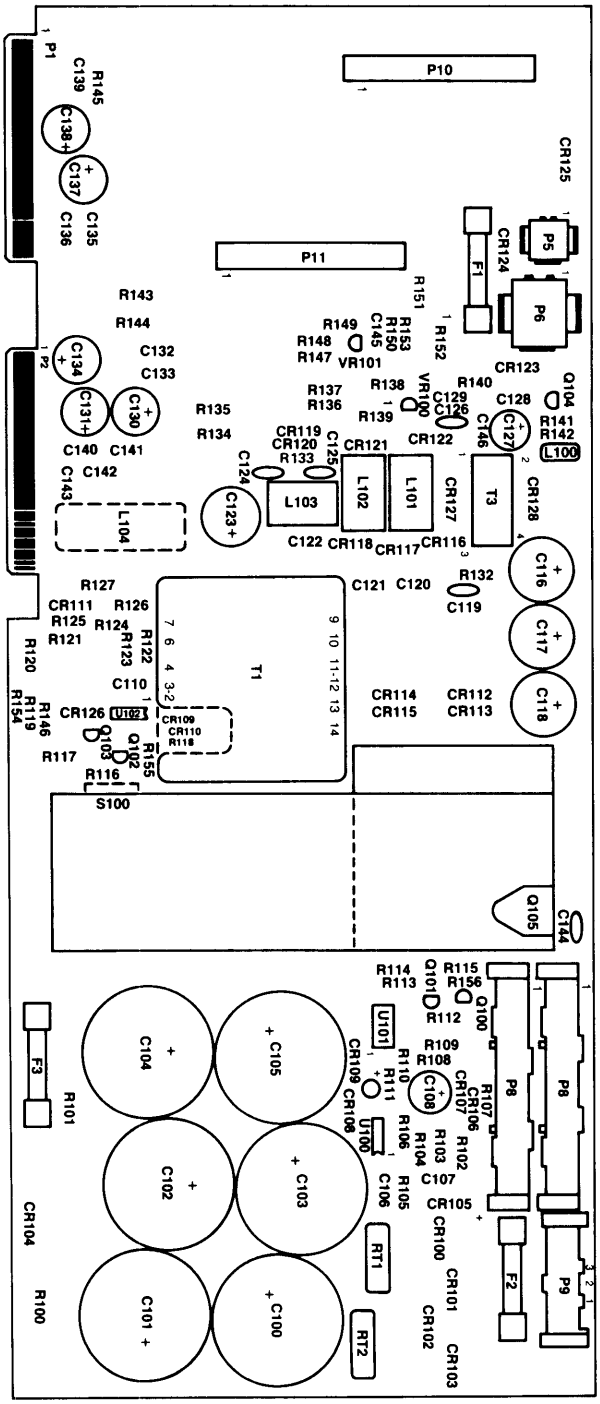


Figure A-4. Parts Locations for Board 1, 440W Supply

Table A-7. 440W Supply Board 1 Replaceable Parts (Sheet 1 of 3)
(HP 0950-1671, Boschert XL0400-5411R)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
001	13713	P C B XL400-5411R	
002			
003	13825	SUB ASSY XFMR WIRING HARNESS BD	T1, (REF)
004	13824	SUB ASSY DIODE HEATSINK XL400-3411	R128, R129, R130, R131, C111, C112, C113, C114, CR112, CR113, CR114, CR115, (REF)
005	13835	BUS BAR RETURN	
006	13836	HEATSINK DIODE	
007			
008			
009			
010	13536	CAP ELECT 1000UF-200V SNAP-IN TERM	C100, C101, C102, C103, C104, C105
011	11659-02	CAP ELECT RAD LDS 3.3UF-250V DC	C108
012	10520-04	OBSOLETE(CAP ELECT 3.3UF-50V)	C109
013	10765-18	CAP ELECT 330UF 63V RD LDS VB	C123
014	11106-01	CAP ELECT R LDS 470UF-25VDC LO ESR	C130, C131
015	11110-01	CAP ELECT 1000UF 10V	C134, C137, C138
016	12951	CAP ELECT 10UF-100VDC LD ESR R LDS	C127
017	2032	CAP CERM .001UF 10% 1000VDC	C119
018	2060	CAP CERM 470PF-1KV	C124, C125, C126
019	10765-06	CAP ELECT 3.3UF 63V RD LDS VB	C109, REPLACES ITEM #012
020	12685-07	CAP MET POLY .47UF 5% 63/100V	C107
021	13594-02	CAP MET POLY .047UF 250V 5% .4"LS	C106, C128
022	12328-05	CAP MET POLY 0.33UF 63VDC 5%	C110, C133, C135
023	12657-07	CAP MET POLY RLD .10UF 10% 250VDC	C122, C139
024	12328-04	CAP MET POLY 0.22UF 63VDC 5%	C120, C121, C129, C132, C136, C140, C141, C142, C143, C146
025	13723-01	CAP ELECT 2200UF-16V RLD LOZ HI RIP	C116, C117, C118
026	13593-07	CAP MET POLY .01UF 630/1000V 5%	C145
027			
028	2105	CAP CERM 100PF-1KV	C144
029			
030	1021	DIODE HI CUR. AX LDS. 400V,6A MR754	CR100, CR101, CR102, CR103
031	10056-01	OBSOLETE(DIODE FAST RECV)	CR117, CR118
032	11380-04	DIODE FULL WAVE BRDG PREP/10868-04	CR105
033	1038	DIODE GEN PUR 1A IN4004 400VDC	CR106
034	12594	DIODE SILICON CASE DO-35,IN4448	CR107, CR109, CR110, CR126
035			
036	1043	DIODE GEN PUR AX LDS 3A 400V MR504	CR123, CR124, CR125
037	12577-02	DIODE RECT FAST RECV.16A 100V	CR116
038	14461-01	DIODE FST RECV 2A 100V AX LDS	CR119, CR120
039	1014	DIODE ZENER 500 MW 5.6V 5% IN5994B	CR104
040	1008	DIODE ZENER 400MW 15V 5% IN965A	CR108
041	11356-15	DIODE ZENER 500MW 8.2V 5% IN5998B	CR111
042	1042	DIODE FST RCV 200V 3A AX LDS MR852	CR121, CR122, CR127, CR128
043	10056-03	DIODE FAST RECV BYW 29-150	CR117, CR118, REPLACES ITEM #031
044			
045			
046	10180-01	FUSE 5A-250V NORMAL-BLOW	F1, F3
047	10865	FUSE 10 AMP 250V (BUSS ABC10)	F2
048	13799	HARNESS 2 PIN CONN	J14
049	13800	HARNESS 3 PIN CONN	J13
050	13801	HARNESS 4 PIN CONN	J12

Table A-7. 440W Supply Board 1 Replaceable Parts (Sheet 2 of 3)
(HP 0950-1671, Boschert XL0400-5411R)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
051			
052	7421	WIRE RED 18 AWG 5" (1/4 X 1/4) UL10	JP7
053	10009-07	WIRE YEL 18 AWG 2.75 LG 1/4 X 1/4	JP3
054	7949	WIRE RED 18 AWG 3 5/8" (1/4 X 1/4)	JP8
055	10372-11	WIRE WHT 22 AWG 3.75" 1/4 X 1/4	JP4
056	10236-04	WIRE BLK 22 AWG 4.25" (1/4 X 1/4)	JP5
057	10008-06	WIRE ORN 18 AWG 3.75" 1/4 X 1/4	JP6
058			
059	14364	INDUCTOR OUTPUT 70A XL400-5411R	L104
060	10799	INDUCTOR 18 UH	L100
061	10899	INDUCTOR 14.4 UH	L101, L102, L103
062	13332	PCB PIN HEADER 4 CKT GOLD PIN	P5
063	13331	PCB PIN HEADER 12 CKT GOLD PIN	P6
064	13333-02	CONN PCB 10 CKT INLINE PIN HEADER	P7, P8
065	13333-01	CONN PCB 3 CKT INLINE PIN HEADER	P9
066	14565-04	CONN PCB 15/30 CONT PRS .125 CTR	P10, P11
067			
068			
069			
070	1016	TRANS PNP CASE TO-92 2N4126	Q104
071	12592	TRANS PNP CASE TO-92,MPS 2907A	Q102
072	12593	TRANS NPN TO-92 CASE MPS2222A	Q100, Q103
073	12591	TRANS NPN CASE TO 92 2N4124	Q101
074	14263-01	TRANS NPN POWER DARLING 10A TIP 140	Q105
075	3124	OBSOLETE(RES 150K OHM 1/4W)	R156
076	3077	OBSOLETE(RES 1.6K OHM 1/4W)	R146
077	3062	OBSOLETE(RES 390 OHM 1/4)	R141
078	3324	RES CF 150K OHM 5% 1/2W	R102
079	3091	OBSOLETE(RES 6.2K OHM 1/4W)	R104
080	3120	OBSOLETE(RES 100K OHM 1/4W)	R105, R119
081	3096	OBSOLETE(RES 10K OHM 1/4W)	R108, R112, R118, R125
082	3078	OBSOLETE(RES 1.8K OHM 1/4W)	R106
083	3122	OBSOLETE(RES 120K OHM 1/4W)	R111
084	3082	OBSOLETE(RES 2.7K OHM 1/4W)	R110
085	3072	RES CF 1K OHM 5% 1/4W	R109, R148, R154
086	3100	OBSOLETE(RES 15K OHM 1/4W)	R113
087	3088	OBSOLETE(RES 4.7K OHM 1/4W)	R115, R120, R121
088			
089	3093	OBSOLETE(RES 7.5K OHM 1/4W)	R126
090	3112	OBSOLETE(RES 47K OHM 1/4W)	R122
091	3108	OBSOLETE(RES 33K OHM 1/4W)	R124
092	10318-76	OBSOLETE(RES 3K OHM 1/4W)	R149
093	3084	OBSOLETE(RES 3.3K OHM 1/4W)	R155, R150
094	3256	RES CF 220 OHM 5% 1/2W	R133
095	10304-41	RES CF 100 OHMS 5% 1/2W	R132
096	3115	OBSOLETE(RES 62K OHM 1/4W)	R151
097	10318-102	OBSOLETE(RES 1.2 OHM 1/4W)	R142
098	3060	OBSOLETE(RES 330 OHM 1/4)	R136, R140
099	3064	OBSOLETE(RES 470 OHM 1/4W)	R138
100	3092	OBSOLETE(RES 6.8K OHM 1/4W)	R153, R147
101	10233-84	RES MET OXIDE 27K OHM 5% 2W	R100, R101
102	10233-48	RES MET OXIDE 820 OHM 5% 2W	R127

Table A-7. 440W Supply Board 1 Replaceable Parts (Sheet 3 of 3)
(HP 0950-1671, Boschert XL0400-5411R)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
103	11313-51	RES MF 3.32K OHMS 1% 1/8W	R137
104	10232-57	RES MET OXIDE 1K OHM 5% 1 W	R117
105	10232-65	RES MET OXIDE 2.2K OHM 5% 1W	R116
106	3080	OBSOLETE(RES 2.2K OHM 1/4W)	R114
107			
108			
109	10522-29	RES WW 12K OHMS 5% 5W	R107
110	3820	RES WW 120 OHM 5% 2W BWH	R134, R135
111	3903	RES WW 10 OHM 5% 5W	R143, R144
112	10966-49	RES WW 1K OHM 10% 2W BWH	R145
113	3943	RES POT 5K VADJ MTURN .5W CERMET	R139
114	10519-16	RES POT 50K OHM VADJ STURN .5W CERM	R103
115	10519-10	RES POT 2K OHM VADJ STURN .5W CERM	R123, R152
116			
117			
118			
119			
120	3938	RES THERMISTOR DISC 5 OHM 15% ST LD	RT1, RT2
121	3911	THERMOSTAT SNAP-ACTING AUTO RESET	S100
122			
123	14159	TRANSFORMER T3 BIFILAR	T3
124			
125			
126			
127	10505	I C LOW POWER DUAL VOLT COMP LM393N	U100, U102
128	11498	I C OPTO-ISOLATOR OP1-1264B	U101
129	1071	I C SHUNT REG TL430	VR100, VR101
130	14381	TIE WRAP PUSH MOUNT	(REF), L104
131	11661-01	SPACER GLASS .225 OD .067 ID .185TH	(REF), R107, RT1, RT2
132	7501	TIE WRAP MEDIUM	(REF), L104
133	7500	TIE WRAP SMALL	
134	13887	INSULATOR	
135	13831	TRANSFORMER SUPPORT	(REF), T1
136	7015	FUSE CLIP PCB TYPE FOR 3AG FUSE	XF1, XF2, XF3
137	10006-12	WIRE BLU 18AWG 3.75"(1/4 X 1/4)	T3(REF)
138	10004-13	WIRE BLK 18 AWG 3.75"(1/4 X 1/4)	T3(REF)
139	10002-04	WIRE RED 18 AWG 5.50 (1/4 X 1/4)	(REF), S100
140	7578	SCREW P H 4-40 X 1/4	(REF), S100
141	7511	SCREW P H 6-32 X 1/2	(REF), H/S, T1
142			
143	7576	SCREW P H 6-32 X 1/4	Q105(REF)
144	7577	WASHER SPLIT LOCK #4	(REF), S100
145	7588	WASHER SPLIT RING LOCK #6	(REF), H/S, T1
146	7506	WASHER FLAT #6	(REF), H/S, T1
147	11828	WASHER METAL	(REF), Q105
148	12560	MTG HDW TO-220 NON CNDCT #4	(REF), CR116, CR117, CR118
149	10726-01	SPACER INSULATED #6 .250 DIA .125LG	
150	13979	STIFFENER 3.00 X 4.50	(REF), C100, C101, C102, C103, C104, C105
151	13980	STIFFENER .75 X 2.20	(REF), C116, C117, C118
152	13981	STIFFENER .50 X 1.00	(REF), C137, C138
153			
154			
155	12459	LABEL FUSE WARNING	(REF), C104
156	12569	LABEL,CSA MARK	
157	7740	LABEL DANGER HIGH VOLTAGE	(REF), C101

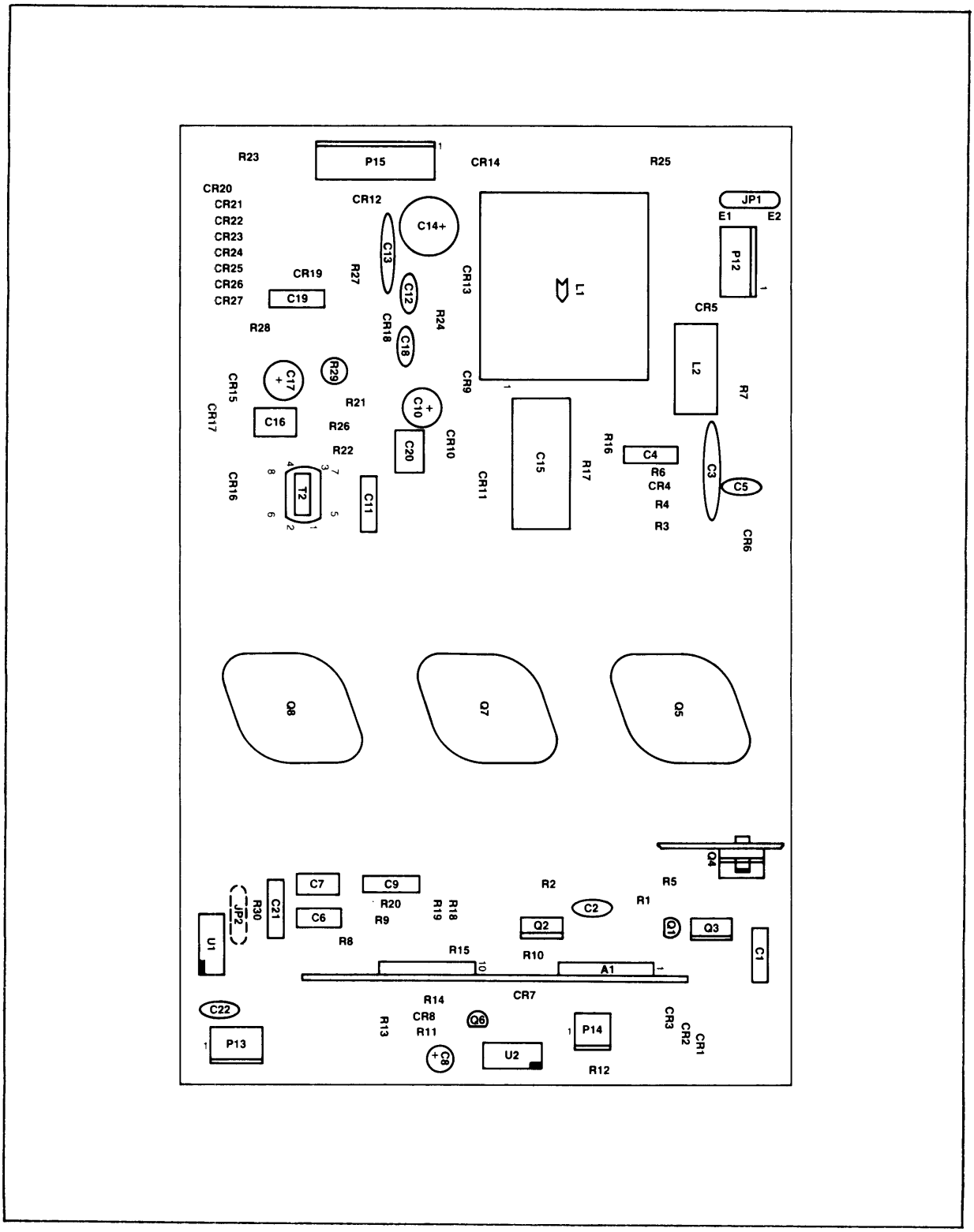


Figure A-5. Parts Locations for Board 2, 440W Supply

Table A-8. 440W Supply Board 2 Replaceable Parts (Sheet 1 of 2)
(HP 0950-1671, Boschert XL0400-5411R)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
001	13619	TRANS NPN POWER 400V 15A 2N6678	Q5, Q7, Q8
002	1118	TRANS NPN POWER MJE13007	Q4
003			
004			
005			
006			
007	3098	OBSOLETE(RES 12K OHM 1/4W)	R30
008	3080	OBSOLETE(RES 2.2K OHM 1/4W)	R8
009	3046	OBSOLETE(RES 82 OHM 1/4W)	R9
010	3096	OBSOLETE(RES 10K OHM 1/4W)	R11
011	3104	OBSOLETE(RES 22K OHM 1/4W)	R10, R20
012	3064	OBSOLETE(RES 470 OHM 1/4W)	R1
013	10318-22	OBSOLETE(RES 16 OHM 1/4W)	R14
014	3048	OBSOLETE(RES 100 OHM 1/4W)	R13
015	3056	OBSOLETE(RES 220 OHM 1/4W)	R12
016	3135	OBSOLETE(RES 430K OHM 1/4W)	R19
017	3040	OBSOLETE(RES 47 OHM 1/4W)	R26
018	3024	OBSOLETE(RES 10 OHM 1/4W)	R6
019	10233-78	RES MET OXIDE 15K OHM 5% 2W	R2, R23
020	13598-08	RES MF UNCUT LDS 365K 1% 1/8W	R18
021	10232-95	RES MET OXIDE 47K OHM 5% 1 W	R21, R22
022	10233-30	RES MET OXIDE 150 OHM 5% 2W	R24, R27
023	10232-16	RES MET OXIDE 20 OHM 5% 1 W	R29
024			
025	3811	RES WW 10 OHM 10% 1W BW20F	R15, R3, R4
026	3816	RES WW .2 OHM 5% 2W BWH	R16, R17
027	12261-13	RES WW 10 OHM 10% 10W	R25
028	3812	RES WW 82 OHM 10% 1W BW20F	R5
029	3918	RES WW 5 OHM 5% 5W	R7
030	10048-65	RES WW 47 OHM 5% 2W BWH	R28
031			
032			
033			
034	12872-01	TRANSFORMER DRIVE,PC MT TWO CORE 4T	T2
035	11498	I C OPTO-ISOLATOR OP1-1264B	U2
036	12975-15	OPTO ISOLATOR SORTED VDE 390 ORN	U1
037	12854	CONN JACK CLOSED ENTRY P C SWAGE MT	E1, E2, E3, E4
038	11661-01	SPACER GLASS .225 OD .067 ID .185TH	(REF), R25
039	11399	ASSY MTG HDW TO-3 CNDCT PEM STUD	(REF), Q5, Q7, Q8
040	7509	SCREW P H 4-40 X 1/2	(REF), Q4, CR6
041	7503	NUT HEX 4-40	(REF), Q4, CR6
042	7602	WASHER INT TOOTH LOCK #4	(REF), Q4, CR6
043	7010	INSULATOR ALUM TO-220	(REF), Q4, CR6
044	11572	INSULATOR TO-220	(REF), Q4, CR6
045	7930	WIRE RED 22 AWG 1 1/2" (1/4, 1/4)T	(REF), T2
046	7202	ORTV 108 CLEAR	
047			
048			
049			
050			
051			
052			

Table A-8. 440W Supply Board 2 Replaceable Parts (Sheet 2 of 2)
(HP 0950-1671, Boschert XL0400-5411R)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
053	13730	P C B	
054	12983	CONTROL BOARD 723	A1
055			
056			
057			
058			
059			
060			
061	13834	HEATSINK TRANSISTOR	(REF), Q5, Q7, Q8
062	10346-01	CAP ELECT 10UF-16V RAD LDS	C8
063	12288	CAP ELECT 4.7UF 20% 100V LO ESR RLD	C10, C17
064	2071	CAP ELECT RAD.LEADS 10UF-250V	C14
065	2105	CAP CERM 100PF-1KV	C2
066	2003	CAP CERM .1UF +80%-20% 500VDC	C3
067	2073	CAP CERM DISC .001UF-3KV 20%	C12, C18
068	2062	CAP CERM 270PF-1KV	C5
069	2032	CAP CERM .001UF 10% 1000VDC	C22
070	2005	CAP CERM DISC .01UF 1KVDC 20% Z5U	C13
071	12328-04	CAP MET POLY 0.22UF 63VDC 5%	C6
072	12328-06	CAP MET POLY 0.47 63VDC 5%	C7
073	13593-05	CAP MET POLY .0047UF 630/1000V 5%	C9, C11, C21
074	12328-08	CAP MET POLY 1.0UF 63VDC 5%	C16, C20
075	13450-01	CAP MET POLY .1UF 100VDC 5% R LDS	C1, C4, C19
076	2080	CAP MET POLY 2UF-200VDC 10%	C15
077			
078			
079			
080			
081	14461-01	DIODE FST RECV 2A 100V AX LDS	CR20, CR21, CR22, CR23, CR24, CR25, CR26, CR27
082	13734	DIODE GEN PUR FST FWD REC IN4004	CR1, CR2, CR3, CR12, CR13, CR18, CR19
083	12594	DIODE SILICON CASE DO-35,IN4448	CR4, CR8
084	1045	DIODE FST RCV 100V 3A AX LDS MR854	CR5, CR14
085	11196	DIODE FAST RECV MR2404F	CR6
086	1028	OBSOLETE(DIODE RECTIFIER)	CR17, CR9
087	12260-04	DIODE HI CUR AX LDS. 30V,6A SR3773	CR10, CR11, CR15, CR16
088	11356-15	DIODE ZENER 500MW 8.2V 5% IN5998B	CR7
089	1155	DIODE FST RCV 1A 600V AX LD IN4937	CR17, CR9, REPLACES ITEM #034
090	10024-02	TERM PLUG .400	JP1
091			
092	13733	INDUCTOR 4MH	L1
093	11625	INDUCTOR 010467-01/02	L2
094			
095			
096	13195-01	CONN FRICTION LOCK 4 POS .156 CTR	P12
097	13195-04	CONN FRICTION LOCK 2 POS .156 CTR	P14
098	13195-05	CONN FRICTION LOCK 3 POS .156 CTR	P13
	13195-06	CONN FRICTION LOCK 7 POS .156 CTR	P15
	12592	TRANS PNP CASE TO-92,MPS 2907A	Q1, Q6
	12675	TRANS PNP HI VOLT 400V TO-126 CASE	Q3
	1005	TRANS NPN POWER TIP-50	Q2

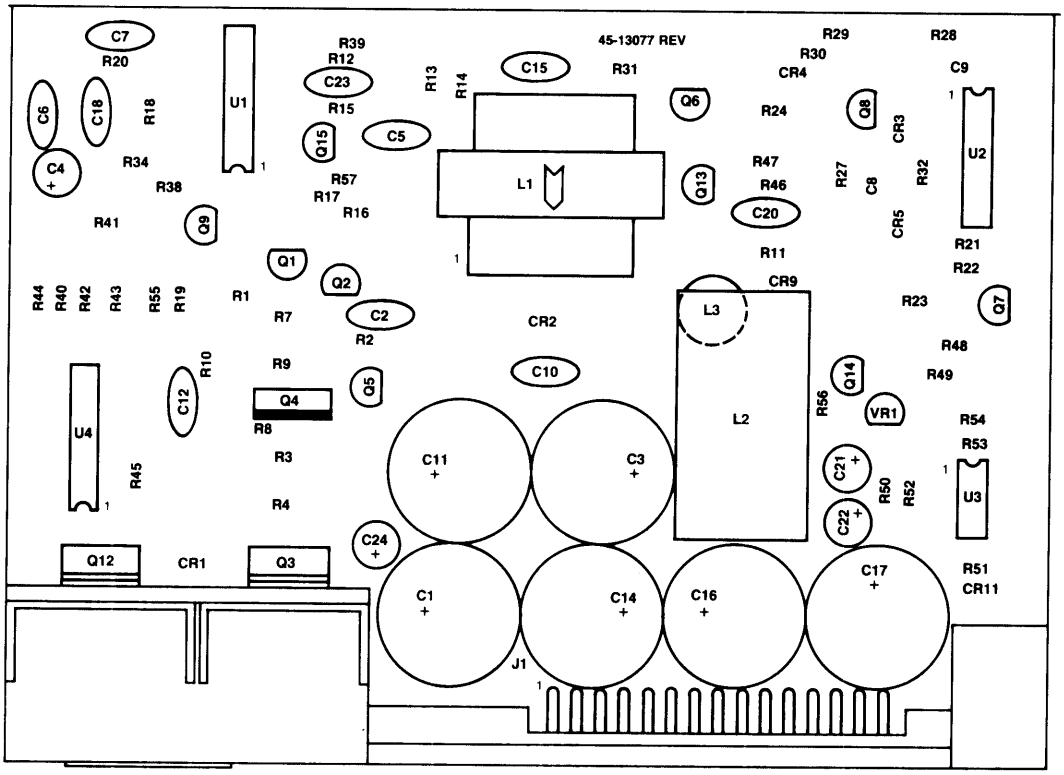


Figure A-6. Parts Locations for Battery Backup Card (BB500)

Table A-9. Battery Backup Card Option Replaceable Parts (Sheet 1 of 3)
(Boschert BB500 used in HP 0950-1671)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
001	13093	P C B	
002			
003			
004	2032	CAP CERM .001UF 10% 1000VDC	C2, C18
005	2008	CAP CERM .1UF-100V	C5, C10, C23
006	2120	CAP CERM .0022UF-1KV	C6, C12
007			
008			
009	2060	CAP CERM 470PF-1KV	C7
010	2062	CAP CERM 270PF-1KV	C15
011	2059	CAP CERM .01UF-100V	C20
012			
013			
014	10520-13	CAP ELECT 470UF 50V RD LDS VB	C1, C3, C11, C14, C16, C17
015	10520-05	OBSOLETE(CAP ELECT 4.7UF-50V)	C4
016	10520-02	OBSOLETED(CAP ELECT 1.0UF-50V)	C21
017	10346-01	CAP ELECT 10UF-16V RAD LDS	C22
018	10520-04	OBSOLETE(CAP ELECT 3.3UF-50V)	C24
019	10765-07	CAP ELECT 4.7UF 63V RD LDS VB	C4, REPLACES ITEM #015
020	13593-01	CAP MET POLY .001UF 630/1000V 5%	C8, C9
021	10765-06	CAP ELECT 3.3UF 63V RD LDS VB	C24, REPLACES ITEM #018
022	10541-03	CAP ALUM ELECT 1.0UF 100W VDC	C21, REPLACES ITEM #016
023	1038	DIODE GEN PUR 1A IN4004 400VDC	CR1
024	12594	DIODE SILICON CASE DO-35,IN4448	CR3, CR4, CR5, CR9, CR11
025			
026	11730-20	HEATSINK SUB ASSY TO-220	CR2, (REF)
027			
028	12540	I C ADJ PREC SHUNT REG TO 92	VR1
029			
030			
031			
032			
033			
034	11969	INDUCTOR 3 TERM REG	L1
035	11762	INDUCTOR 011749	L2
036	12209	INDUCTOR AIR CORE	L3
037			
038			
039	11463	TRANS MPS-A56	Q1, Q2, Q5, Q9
040	11689	TRANS PNP HIGH VOLT D45C11	Q4
041	13530	TRANS NPN POWER 45V 15A D44VH4	Q3
042			
043			
044	11464	TRANS NPN MPS-A06	Q6, Q7, Q15
045	1017	TRANS NPN GEN PUR MPS-5172	Q8
046	1144	TRANS NPN POWER SIL TIP-31 TO-220	Q12
047			
048			
049	12593	TRANS NPN TO-92 CASE MPS2222A	Q13, Q14
050			
051			
052			

Table A-9. Battery Backup Card Option Replaceable Parts (Sheet 2 of 3)
(Boschert BB500 used in HP 0950-1671)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
053			
054			
055	3074	OBSOLETE(RES 1.2K OHM 1/4W)	R1, R7, R40
056	3067	OBSOLETE(RES 620 OHM 1/4W)	R2
057	3048	OBSOLETE(RES 100 OHM 1/4W)	R8
058			
059			
060	3064	OBSOLETE(RES 470 OHM 1/4W)	R10
061			
062	3113	OBSOLETE(RES 51K OHM 1/4W)	R15
063			
064			
065	3120	OBSOLETE(RES 100K OHM 1/4W)	R17, R28
066	3103	OBSOLETE(RES 20K OHM 1/4W)	R55
067	3116	OBSOLETE(RES 68K OHM 1/4W)	R20
068	3110	OBSOLETE(RES 39K OHM 1/4W)	R53
069			
070	3096	OBSOLETE(RES 10K OHM 1/4W)	R21, R22, R45, R57, R47
071	3072	RES CF 1K OHM 5% 1/4W	R23, R29, R42
072	3091	OBSOLETE(RES 6.2K OHM 1/4W)	R24
073	3092	OBSOLETE(RES 6.8K OHM 1/4W)	R46
074			
075	3090	OBSOLETE(RES 5.6K OHM 1/4W)	R27, R50
076	3108	OBSOLETE(RES 33K OHM 1/4W)	R30
077	3134	OBSOLETE(RES 390K OHM 1/4W)	R32, R34
078			
079			
080	3087	OBSOLETE(RES 4.3K OHM 1/4W)	R38
081			
082			
083	3040	OBSOLETE(RES 47 OHM 1/4W)	R39
084	10318-05	OBSOLETE(RES 3.3 OHM 1/4W)	R44
085	3068	OBSOLETE(RES 680 OHM 1/4W)	R52
086			
087			
088	3088	OBSOLETE(RES 4.7K OHM 1/4W)	R54, R56
089	3084	OBSOLETE(RES 3.3K OHM 1/4W)	R51
090			
091	10329-91	RES MF 43K 2% 1/4W	R12
092	10329-61	RES MF 2.4K 2% 1/4W	R16
093	10329-03	RES MF 270K 2% 1/4W	R31
094	10329-74	RES MF 8.2K 2% 1/4W	R14
095			
096			
097	3094	OBSOLETE(RES 8.2K OHM 1/4W)	R43, R19
098	3102	OBSOLETE(RES 18K OHM 1/4W)	R48
099			
100			
101			
102			
103			
104	3800	RES WW .1 OHM 5% 2W BWH	R3, R4

Table A-9. Battery Backup Card Option Replaceable Parts (Sheet 3 of 3)
(Boschert BB500 used in HP 0950-1671)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
105	3811	RES WW 10 OHM 10% 1W BW20F	R9
106	10966-49	RES WW 1K OHM 10% 2W BWH	R11
107			
108	10660-10	RES POT 2K OHMS VERT ADJ .5W CERM	R13
109	3902	RES POT 5K OHM HORZ ADJ 10% .75W	R41, R49
110	10519-10	RES POT 2K OHM VADJ STURN .5W CERM	R18
111			
112			
113	1000	I C VOLT REG 723	U1, U4
114	10379-01	I C NOR GATE DUAL IN CMOS CD4001BE	U2
115	10505	I C LOW POWER DUAL VOLT COMP LM393N	U3
116			
117	13495	BRACKET BATT BACK-UP XL400-3502	
118	7577	WASHER SPLIT LOCK #4	
119	7506	WASHER FLAT #6	
120	7588	WASHER SPLIT RING LOCK #6	
121	11092-01	STANDOFF HEX 4-40 .250LG	
122	13744	MTG HDW TO-220 NON CNDCT #4	
123	14578	LABEL SMALL BOSCHERT MODEL/SER NO	
124	13570	HEATSINK TO-220 NO MTG TABS	
125	15178	LABEL CUSTOMER ID	L1(REF)
126	7863	0.0400TAPE 2 SIDED 1/16	L2, (REF)
127			
128	12891-03	PEM STUD 4-40 THREAD .625 LG	
129	13618-01	SCREW CAPTIVE PANEL .625" LG 6-32	

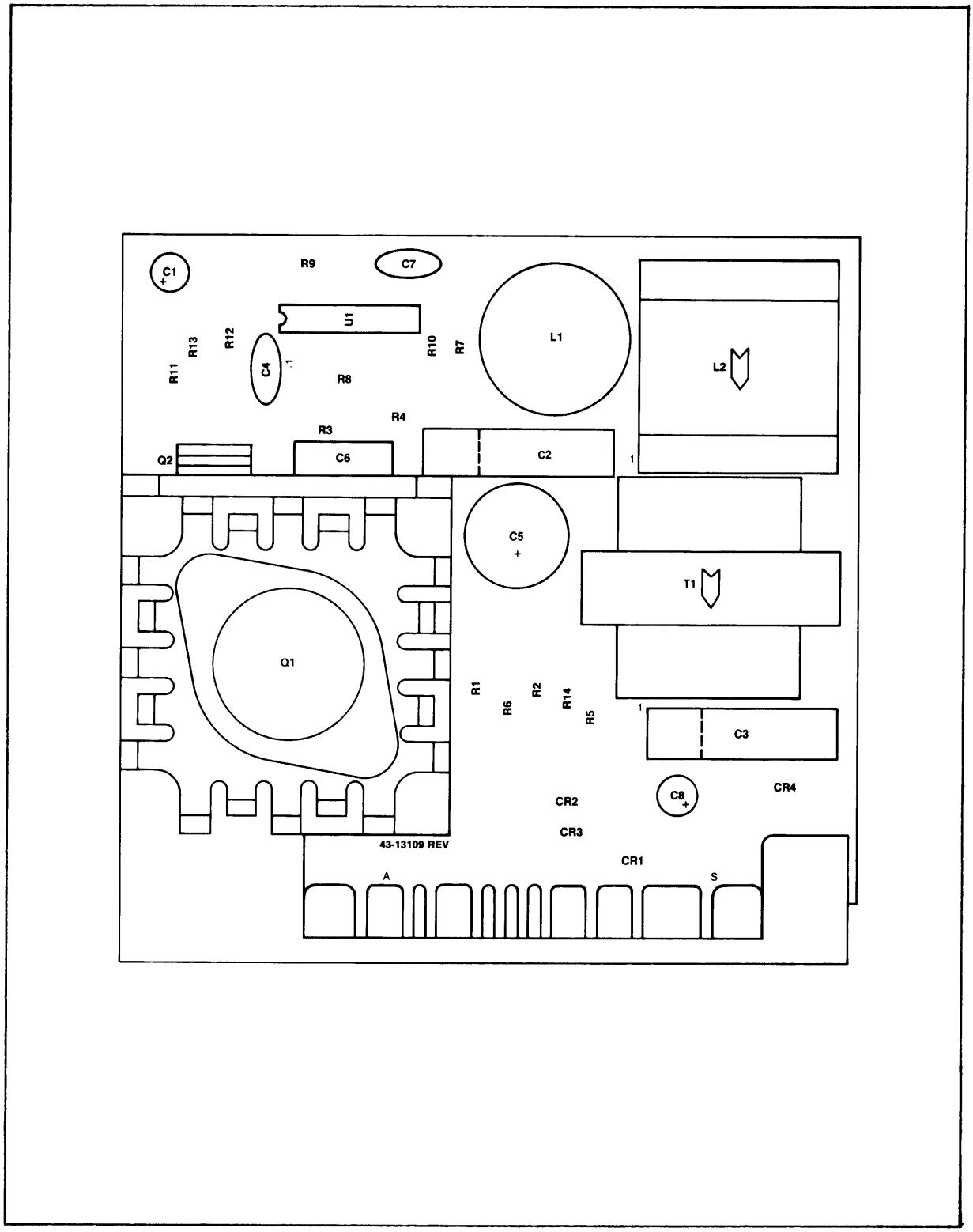


Figure A-7. Parts Locations for 25 kHz Card (SW100)

Table A-10. 25 kHz Card Option Replaceable Parts (Sheet 1 of 2)
(Boschert SW100 used in HP 0950-1671)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
001	13137	P C B SW100	
002	13527	HEATSINK TO-3	
003	13494	BRACKET SINE WAVE,SW100	
004			
005	2059	CAP CERM .01UF-100V	C4
006	11133-01	CAP MET POLY .1UF 100V 10% RLDS	C6
007			
008			
009			
010	13249-01	OBSOLETE(CAP ELECT 100UF-50V)	C5
011	14297-04	CAP ELECT 100UF 50V LO ESR RLDS RX	C5, REPLACES ITEM #010
012	10765-06	CAP ELECT 3.3UF 63V RD LDS VB	C1, C8
013	2032	CAP CERM .001UF 10% 1000VDC	C7
014			
015			
016	14580-01	CAP MET POLYP 0.22UF 250V 5% RLD	C2
017	14580-02	CAP MET POLYP 0.33UF 250V 5% RLD	C3
018			
019			
020	1088	OBSOLETE(DIODE FST RCV 100V-3A)	CR1, CR2
021	1026	OBSOLETE(DIODE 200V-1A)	CR3, CR4
022	1155	DIODE FST RCV 1A 600V AX LD IN4937	CR3, CR4, REPLACES ITEM #021
023	1042	DIODE FST RCV 200V 3A AX LDS MR852	CR1, CR2, REPLACES ITEM #020
024			
025	13163	INDUCTOR ASSY 140UH @ 1.75 ADC	L1
026	13164	INDUCTOR 100 UH @ 1.25A PQ26/25	L2
027			
028			
029	1094	TRANS NPN POWER 2N5885	Q1
030	1144	TRANS NPN POWER SIL TIP-31 TO-220	Q2
031			
032			
033			
034	3040	OBSOLETE(RES 47 OHM 1/4W)	R3
035	3082	OBSOLETE(RES 2.7K OHM 1/4W)	R9, R5
036	3068	OBSOLETE(RES 680 OHM 1/4W)	R4
037			
038			
039	3087	OBSOLETE(RES 4.3K OHM 1/4W)	R14
040	3100	OBSOLETE(RES 15K OHM 1/4W)	R13
041	3104	OBSOLETE(RES 22K OHM 1/4W)	R8
042			
043			
044	3113	OBSOLETE(RES 51K OHM 1/4W)	R11
045	3052	OBSOLETE(RES 150 OHM 1/4W)	R6
046	3092	OBSOLETE(RES 6.8K OHM 1/4W)	R10
047	3272	RES CF 1K OHM 5% 1/2W	R7
048			
049			
050	10048-02	RES WW .36 OHM 5% 2W BWH	R1, R2
051			
052	3902	RES POT 5K OHM HORZ ADJ 10% .75W	R12

Table A-10. 25 kHz Card Option Replaceable Parts (Sheet 2 of 2)
(Boschert SW100 used in HP 0950-1671)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
053			
054	14578	LABEL SMALL BOSCHERT MODEL/SER NO	
055			
056	13162	TRANSFORMER DRIVE XL400	T1
057			
058	13618-01	SCREW CAPTIVE PANEL .625" LG 6-32	
059			
060	1000	I C VOLT REG 723	U1
061	11247-19	SCREW P H 6-32 X 1.5 LG	L1, (REF)
062	7504	NUT HEX 6-32	L1, (REF)
063	7506	WASHER FLAT #6	
064	7549	WASHER FENDER	L1, (REF)
065			
066	7580	INSULATOR MICA TO-3	Q1, (REF)
067			
068	7588	WASHER SPLIT RING LOCK #6	
069	11828	WASHER METAL	Q2, (REF)
070	12298	INSULATOR TO-220	Q1, Q2, (REF)
071	7505	WASHER FLAT #4	Q1, Q2, (REF)
072	7577	WASHER SPLIT LOCK #4	
073	7584	SCREW F H 4-40 X 1/2	Q2, (REF)
074	11092-01	STANDOFF HEX 4-40 .250LG	
075	11837	INSULATOR CHOMERICS	Q2, (REF)

A.3 300-WATT SUPPLY

The 300-watt supply, Part No. 0950-1646 is used with the 16-slot backplane for the Micro/1000 computers. The supply operates from either 115 Vac or 230 Vac. There are four fans for cooling (two fans blow across the supply and two fans blow through the I/O card cage). The fans plug into either connector J3 for 115 Vac operation or into J2 for 230 Vac operation

The supply has four dc outputs at +5V, +12V, -12V and +28V (backup battery source). The power supplies for the battery backup (BB) 5M voltage and the 25 kHz ac outputs are provided by separate optional cards that plug into the backplane. The main supply generates a 25 kHz square wave that is input to the 25 kHz sine-wave card as the source for its sine wave output. The 25 kHz sine wave is used as a power source for certain I/O cards. A block diagram of the 300-Watt supply is shown in Figure A-8.

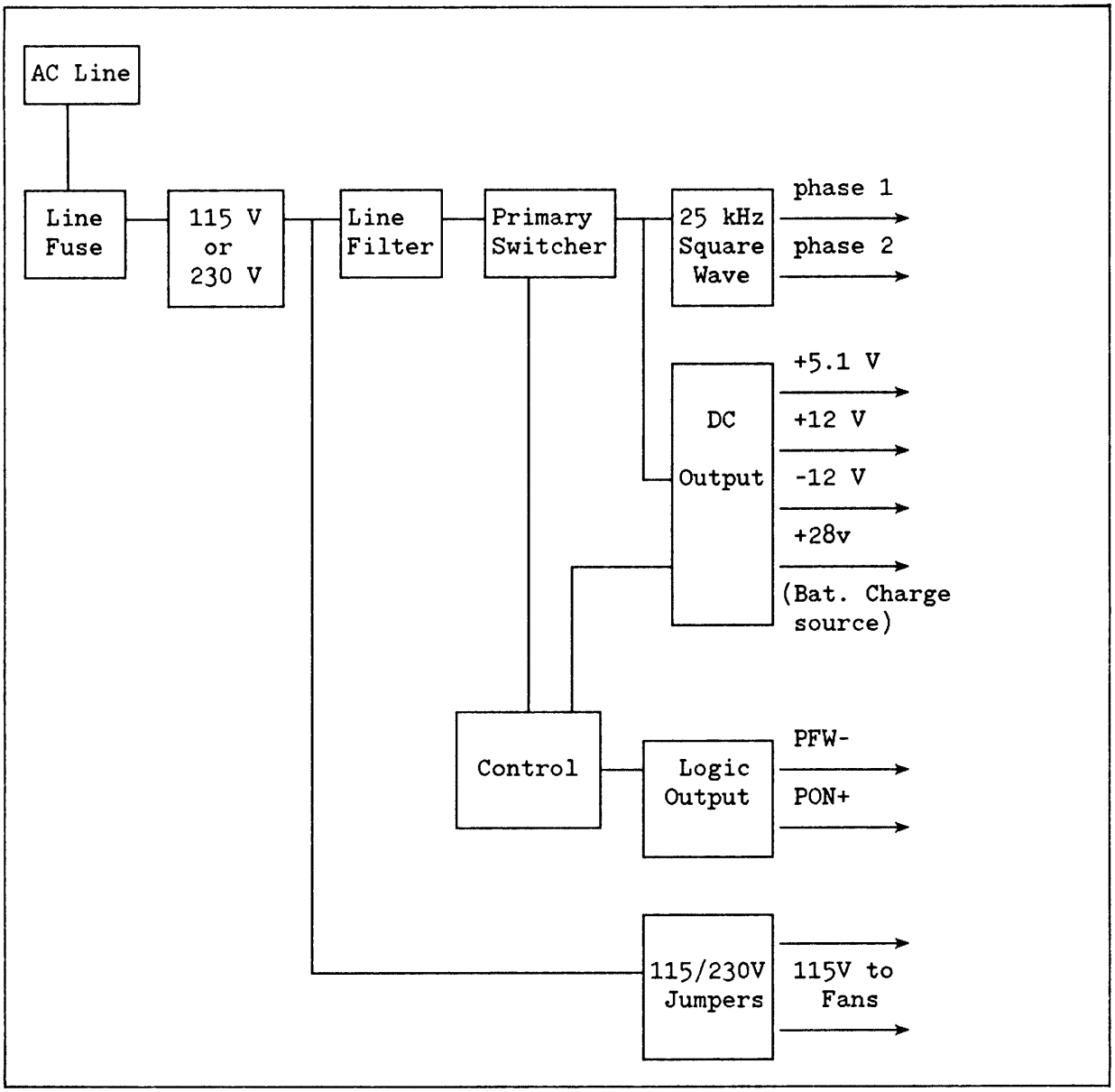


Figure A-8. 300-Watt Power Supply Block Diagram

A.3.1 LOGICAL SIGNALS

The power supply logical control signals to the computer are the same as for the 440 watt supply described in the previous subsection, except that the MLOST- signal goes directly from the battery backup card to the backplane rather than through the power supply as in case of the 440 watt supply.

A.3.2 MECHANICAL SPECIFICATIONS (300W SUPPLY)

The overall mechanical dimensions and connector locations of the 0950-1646 supply are shown in Figure A-9. The power supply connectors are shown schematically in Figure A-10. The connector specifications are given in Table A-11.

The cooling air flow should be a minimum of 40 CFM of air flowing across the power board in the direction indicated in Figure A-9. Power supply assemble diagrams are provided at the rear of this section of the manual.

Table A-11. Connector Specifications (300W Supply)

CONNECTOR	PART NO.
P1	4-582390-4 (AMP)
J2	207378-1 (AMP)
J3	207378-1 (AMP)
J4	EAC-303 (SWITCHCRAFT)

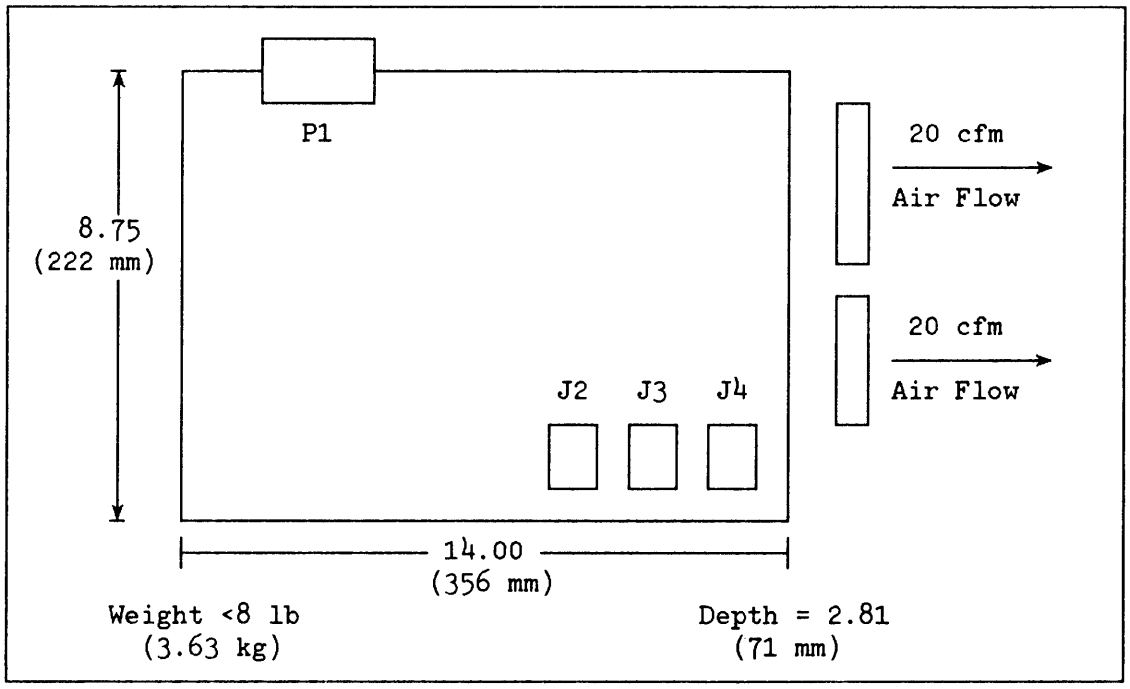


Figure A-9. Dimensions and Connector Locations (300W Supply)

A.3.3 ELECTRICAL CONNECTIONS (300W SUPPLY)

The electrical contacts for the 300 watt power supply are provided by a plug and three jacks. The plug inserts into a jack on the backplane and the jacks are for the fans and ac input. A power supply connector diagram is shown in Figure A-10. The electrical connector definitions are given in Table A-12.

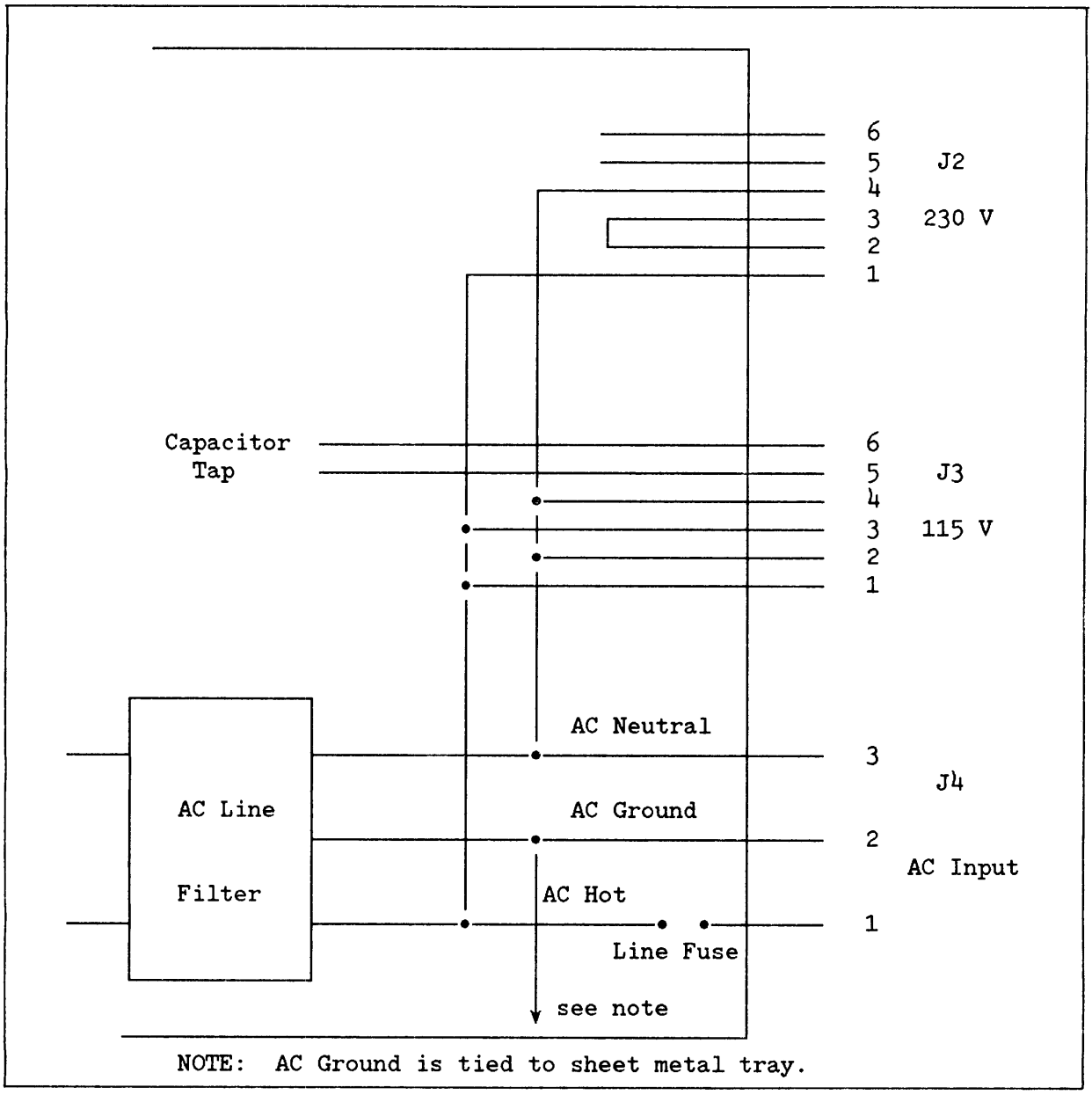


Figure A-10. 300-Watt Power Supply Connector Diagram

Table A-12. Electrical Connections (300W Supply)

ELECTRICAL CONNECTOR PIN DEFINITIONS	
P1 DC OUTPUT CONNECTOR	
Pin Number	**Signal Name**
24 thru 35	+5.1 Volts
10 thru 23	Common
6 thru 8	+12 Volts
5	-12 Volts
9	+28 Volts
3	25 KHz Phase 1
4	25 KHz Phase 2
1	PON+
2	PFW-
J4 AC LINE INPUT	
Pin Number	**Signal Name**
1	AC Line
2	AC Ground
3	AC Neutral
Pin 2 of this input connector must be tied to the sheet metal base.	
J2/J3 AC LINE CONFIGURATION / FANS	
Pin Number	**Signal Name**
1	Fan #1
2	Fan #1
3	Fan #2
4	Fan #2
5	230V / 115V
6	230V / 115V

A.3.4 ELECTRICAL SPECIFICATIONS (300W SUPPLY)

The electrical specifications of the 300 watt supply are provided in the tables below. Ac line input specifications are given in Table A-13, and supply output specifications are given in Table A-14.

Table A-13. Input Electrical Specifications (300W Supply)

AC LINE SPECIFICATIONS

These specifications do not include the power required for the fans.

	Min	Nominal	Max	
Range 1				
Voltage	84	115	140	Volts RMS
RNS Current (Max)	5.7	4.5	4.3	Amps
Inrush	-	-	102	Amps
Range 2				
Voltage	168	230	280	Volts RMS
RMS Current (Max)	2.8	2.3	2.1	Amps
Inrush	-	-	204	AMps
Carry Over	16.0	-	-	mSec
PFW Trip Point				
Range 1	-	-	84	Volts RMS
Range 2	-	-	168	Volts RMS
Line Frequency	47	60	67	Hz
Line Fuse	-	-	10.0	Amps
Input Power (Not including fans)	-	-	400	Watts

Notes:

1. The line filter reduces conducted noise to 2 dBμV below the VDE 0871 Level B conducted emission specification. Conducted noise is measured with the power supply configured for 230 Vac operation and its output loaded for 300W by a resistive load. For 115 Vac operation of the supply, conducted noise must be 2 BμV below the FCC Level B conducted emission specification.

2. Power supply operation permits input transients of up to 3000V for periods of not less than 10 μs.

Table A-14. Output Electrical Specifications (300W Supply)

Maximum Dynamic Load Change: 10% over 10 microseconds

Output Stress Conditions Allowed:

a. Supply will recover from a short to ground or to another regulated output and excessive ambient temperature.
b. Over rated operated temperature.

Output Regulation:

Note: For the following specifications to be valid Output #1 will have at least a 3.0 amp load. With a load less than 3.0 amps on Output #1, the maximum ripple on outputs #1, 2, 3, 4, and 6 must not exceed 1.5 volts peak to peak.

Output # 1	Nominal Voltage	5.1	Volts
	Maximum Current	50	Amps
Regulation	0.0 to 3.0 Amps	+10%	-10%
	3.0 to 6.2 Amps	+5%	-5%
	6.2 to 50.0 Amps	+2%	-2%
Ripple	Maximum Ripple	0.10	Volt
Output # 2	Nominal Voltage	12.0	Volts
	Maximum Current	7.0	Amps
Regulation	0.0 to .03 Amps	+10%	-10%
	.03 to 7.0 Amps	+6%	-3%
Turn-on Surge	9.0 Amps (for 10 sec. max.)	+6%	-30%
Ripple	Maximum Ripple	0.12	Volt
Output # 3	Nominal Voltage	-12.0	Volts
	Maximum Current	3.0	Amps
Regulation	0.0 to .10 Amps	+12%	-12%
	.10 to 3.0 Amps	+6%	-6%
Ripple	Maximum Ripple	0.12	Volt
Output # 4	Square Wave		

The Square Wave outputs must be in regulation when the load on +5.1 Volts is greater than 6.2 Amps.

	Min	Nominal	Max	
Phase to Phase	11.20	11.87	12.54	Volts RMS
Phase to Ground	5.60	5.94	6.27	Volts RMS
RMS Current	0.00	-	3.30	Amps/Phase
Frequency	24k	28k	32k	Hertz
Output Power	-	-	36	Watts

Table A-14. Output Electrical Specifications (300W Supply) (Continued)

Output # 5	Fan Power		
	Nominal Voltage	115	Volts RMS
	Maximum Current	1.25	Amps
Output # 6	Nominal Voltage	+28.0	Volts
	Maximum Current	2.50	Amps
Regulation	0.0 to 2.5 Amps	+20%	-20%
Ripple	Maximum Ripple	0.30	Volt

Note:
Although the sum of the maximums listed above exceeds the 300 Watt specification of the power supply front end, not all of the outputs will be at maximum load at the same time and the actual power is 325 watts for a maximum of 10 seconds.

A.3.5 ENVIRONMENTAL SPECIFICATION (300W SUPPLY)

The environmental specifications of the 0950-1671 300-watt supply are the same as for the 440-watt supply described in the subsection A.2 above.

A.3.6 REPLACEABLE PARTS (300W SUPPLY)

Replaceable parts lists for the 300W power supply are provided in Table A-15 and they may be located in Figure A-11. The table and figures are located in the back of this section in front of the schematics.

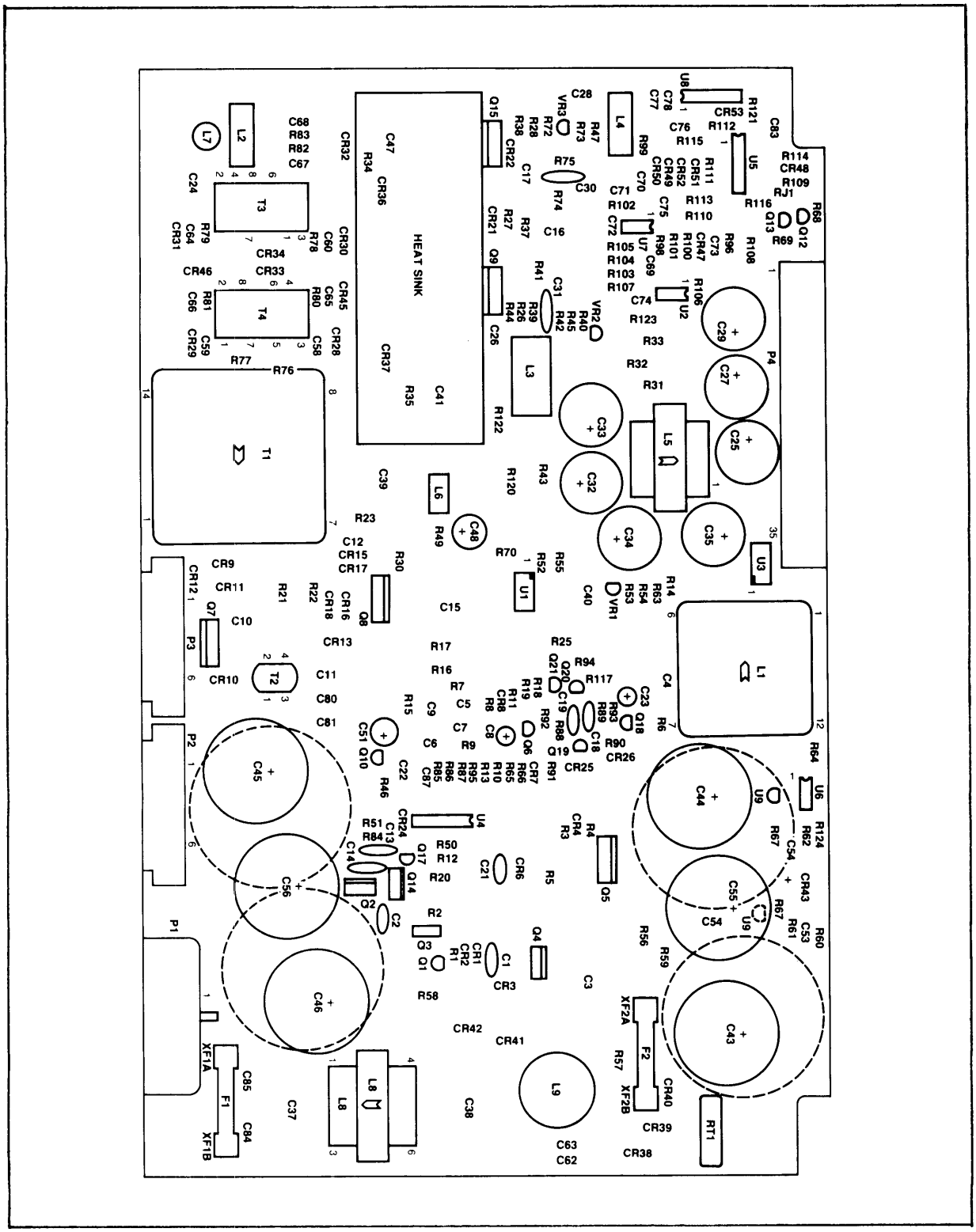


Figure A-11. Parts Locations for 300W Supply

Table A-15. 300W Supply Replaceable Parts (Sheet 1 of 5)
(HP 0950-1646, Boschert XL0301-5612)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
001	14645	P C B XL301-5612	
002	14718	BRACKET L 300 WATT	
003	15009	STUD,LOCATING	
004	14732	HEATSINK XL301-5612	
005	15632-01	HEATSINK	Q5, S1, (REF)
006	15632-02	HEATSINK	Q7, Q8, (REF)
007	12012	HEATSINK TO-220	CR6, Q4, (REF)
008	14995	PLATE COVER	
009	2105	CAP CERM 100PF-1KV	C2
010	2000	CAP CERM .0047UF-1KV	C13, C14
011	2059	CAP CERM .01UF-100V	C18
012	2006	CAP CERM 150PF-1KV	C19
013	2008	CAP CERM .1UF-100V	C1, C41, C47
014	2062	CAP CERM 270PF-1KV	C21
015	2004	CAP CERM .1UF-16V	C30, C31
016			
017	15111-01	CAP SOLID ELECT 6.8uf 20% 25V RLDS	C11, C80, C81
018			
019	10346-05	CAP ELECT 47UF-16V RAD LDS	C51
020	10346-01	CAP ELECT 10UF-16V RAD LDS	C8
021	14992-01	CAP ELEC 1600UF 15V COMP GRLDS	C35
022	14993-01	CAP ELEC 570UF 40V COMP GRLDS	C27, C29, C32, C33, C34
023	15869-05	CAP ELEC 200V 1200UF 85deg C SNP IN	C43, C44, C45, C46, C55, C56
024	12939	CAP ELECT LO LEAK R LDS 100UF 10VDC	C23
025	14994-01	CAP ELEC 350UF 60V COMP GRLDS	C25
026			
027			
028	12951	CAP ELECT 10UF-100VDC LD ESR R LDS	C48
029			
030	15032-10	CAP F&F POLYC 0.68UF 400V 10% RLDS	C3
031	13593-07	CAP MET POLY .01UF 630/1000V 5%	C10
032	12685-03	CAP MET POLY 0.1UF 5% 63/100V	C4, C6, C69, C70, C71, C72, C73, C74, C75, C76
033	13279-07	CAP POLY RLD BOX .10UF 10% 250V	C5, C24, C39
034	12328-04	CAP MET POLY 0.22UF 63VDC 5%	C16, C17, C26, C28, C40
035	12685-07	CAP MET POLY .47UF 5% 63/100V	C7
036	13593-03	CAP MET POLY .0022UF 630/1000V 5%	C9, C87
037			
038	12328-08	CAP MET POLY 1.0UF 63VDC 5%	C83
039	11133-01	CAP MET POLY .1UF 100V 10% RLDS	C12, C53
040	2080	CAP MET POLY 2UF-200VDC 10%	C15
041			
042	15016-07	CAP F&F POLYC 1000PF 160VDC/100VAC	C58, C59, C60, C64, C65, C66, C67, C68
043	11133-03	CAP MET POLY .22UF 100V 10% RLDS	C22
044	12685-10	CAP MET POLY .022UF 5% 63/100V	C77, C78
045	13932-03	CAP MET EMI SUP X-CAP 1.00UF 250VAC	C37, C38
046			
047			
048			
049			
050	2021	CAP TANT 47UF-6V	C54
051	11414-02	CAP MET PAPER .0022UF 250VAC VDE AP	C62, C63, C84, C85
052	1038	DIODE GEN PUR 1A IN4004 400VDC	CR1, CR2, CR3, CR9, CR10, CR11, CR12

Table A-15. 300W Supply Replaceable Parts (Sheet 2 of 5)
(HP 0950-1646, Boschert XL0301-5612)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
053	12594	DIODE SILICON CASE DO-35,IN4448	CR8, CR25, CR26, CR49, CR50, CR51, CR52, CR53
054	14990-01	DIODE FST RCV 9A 400V BYV29-400	CR6
055	10056-03	DIODE FAST RECV BYW 29-150	CR28, CR29
056			
057	10672	DIODE SCHOT STD POL 45V 60A SD-51	CR36, CR37
058	14482	DIODE DUAL SCHOT 20A 45V TO-220	CR32
059	12260-04	DIODE HI CUR AX LDS. 30V,6A SR3773	CR13
060	1021	DIODE HI CUR. AX LDS. 400V,6A MR754	CR38, CR39, CR41, CR42
061			
062			
063			
064	15088-01	DIODE FST RCV 3.5A 150V AXLD	CR33, CR34
065	14461-01	DIODE FST RECV 2A 100V AX LDS	CR4, CR15, CR16, CR17, CR18, CR30, CR31, CR45, CR46
066			
067	11356-15	DIODE ZENER 500MW 8.2V 5% IN5998B	CR7
068	1014	DIODE ZENER 500 MW 5.6V 5% IN5994B	CR40
069	10879	DIODE ZENER 400MW 68V 5% IN5266B	CR21, CR22
070	1008	DIODE ZENER 400MW 15V 5% IN965A	CR24
071	1056	DIODE ZENER 500MW 5.1V 5% IN5993B	CR43, CR47, CR48
072			
073			
074	7015	FUSE CLIP PCB TYPE FOR 3AG FUSE	XF1A, XF1B, XF2A, XF2B
075	10180-01	FUSE 5A-250V NORMAL-BLOW	F2
076	10865	FUSE 10 AMP 250V (BUSS ABC10)	F1
077			
078	12472	RECEPTACLE AC RT ANGLE 6A 250V	P1
079	15001-01	CONN IN LINE RT ANG PIN HDR 6 POS	P2, P3
080	14987-01	CONN RT ANG PCB 35 POS	P4
081			
082	13733	INDUCTOR 4MH	L1
083	14033	INDUCTOR PWDR IRON 3.4UH @ 6A	L4
084	14034	INDUCTOR PWDR IRON 5UH 10A	L3
085	15115	INDUCTOR 5V	L5
086	10799	INDUCTOR 18 UH	L6
087	14850	BALUN SUPER E-375 XL301	L8
088	15012	BALUN TOROIDAL 2x1mh XL301	L9
089	12209	INDUCTOR AIR CORE	L7
090	11944	INDUCTOR 18UH MIN.	L2
091			
092	12592	TRANS PNP CASE TO-92,MPS 2907A	Q1, Q6, Q18
093	1005	TRANS NPN POWER TIP-50	Q2, Q14
094	12675	TRANS PNP HI VOLT 400V TO-126 CASE	Q3
095			
096	1118	TRANS NPN POWER MJE13007	Q4
097			
098	1016	TRANS PNP CASE TO-92 2N4126	Q17
099	14988-02	TRANS NPN POWER 15A 1000V BUW 13A	Q5, Q7, Q8
100	14263-01	TRANS NPN POWER DARLING 10A TIP 140	Q9, Q15
101	13595	TRANS N-CHANNEL POWER FET VN10KM	Q12, Q13
102	12591	TRANS NPN CASE TO 92 2N4124	Q10, Q19, Q20, Q21
103			
104			

Table A-15. 300W Supply Replaceable Parts (Sheet 3 of 5)
(HP 0950-1646, Boschert XL0301-5612)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
105			
106	3081	OBSOLETE(RES 2.4K OHM 1/4W)	R14, R54
107	3124	OBSOLETE(RES 150K OHM 1/4W)	R50, R51
108	3108	OBSOLETE(RES 33K OHM 1/4W)	R65, R88, R90
109	3099	OBSOLETE(RES 13K OHM 1/4W)	R66
110	3088	OBSOLETE(RES 4.7K OHM 1/4W)	R85
111	3072	RES CF 1K OHM 5% 1/4W	R86, R87
112	3110	OBSOLETE(RES 39K OHM 1/4W)	R84
113			
114			
115	10318-47	OBSOLETE(RES 180 OHM 1/4W)	R27, R28
116	3064	OBSOLETE(RES 470 OHM 1/4W)	R1, R53
117	3024	OBSOLETE(RES 10 OHM 1/4W)	R6, R34, R35, R76, R77, R78, R79, R80, R81, R82, R83
118	3098	OBSOLETE(RES 12K OHM 1/4W)	R7, R117
119	3080	OBSOLETE(RES 2.2K OHM 1/4W)	R8, R30, R67, R60
120			
121	3046	OBSOLETE(RES 82 OHM 1/4W)	R9
122	3104	OBSOLETE(RES 22K OHM 1/4W)	R10
123	3096	OBSOLETE(RES 10K OHM 1/4W)	R11, R45, R47, R63, R68, R69, R93, R100, R101, R116, R124
124			
125	3048	OBSOLETE(RES 100 OHM 1/4W)	R13
126	3020	OBSOLETE(RES 6.8 OHM 1/4W)	R95
127	3135	OBSOLETE(RES 430K OHM 1/4W)	R19
128			
129			
130	3128	OBSOLETE(RES 220K OHM 1/4W)	R62
131	3069	OBSOLETE(RES 750 OHM 1/4W)	R64
132			
133			
134	3092	OBSOLETE(RES 6.8K OHM 1/4W)	R40, R75
135	3089	OBSOLETE(RES 5.1K OHM 1/4W)	R109, R114
136	10318-102	OBSOLETE(RES 1.2 OHM 1/4W)	R49
137	3084	OBSOLETE(RES 3.3K OHM 1/4W)	R39, R73
138			
139	3106	OBSOLETE(RES 27K OHM 1/4W)	R37, R42
140	3100	OBSOLETE(RES 15K OHM 1/4W)	R46
141	3079	OBSOLETE(RES 2.0K OHM 1/4W0	R103, R104, R112
142			
143			
144	3060	OBSOLETE(RES 330 OHM 1/4)	R26, R38, R52
145	3120	OBSOLETE(RES 100K OHM 1/4W)	R89, R111
146	3130	OBSOLETE(RES 270K OHM 1/4W)	R92
147	3122	OBSOLETE(RES 120K OHM 1/4W)	R91
148			
149	10328-34	OBSOLETE(RES 680K OHM 1/4W)	R102, R105
150	10328-36	OBSOLETE(RES 820K 1/4W)	R110
151	10328-45	OBSOLETE(RES 2M OHM 1/4W)	R121
152			
153	10232-17	RES MET OXIDE 22 OHM 5% 1 W	R23
154	3336	RES CF 470K OHM 5% 1/2W	R12
155	10304-71	RES CF 1.8K 5% 1/2W	R44, R72

Table A-15. 300W Supply Replaceable Parts (Sheet 4 of 5)
(HP 0950-1646, Boschert XL0301-5612)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
156			
157			
158	11313-01	RES MF 1.0K OHMS 1% 1/8W	R94
159	11891-6491	RES MF 6.49K OHMS 1% 1/4W	R61
160	13598-08	RES MF UNCUT LDS 365K 1% 1/8W	R18
161	11891-4873	RES MF 487K OHMS 1% 1/4W	R59
162	10318-88	OBSOLETE(RES 9.1K OHM 1/4W)	R98
163			
164			
165	10966-49	RES WW 1K OHM 10% 2W BWH	R31
166	10232-95	RES MET OXIDE 47K OHM 5% 1 W	R20
167			
168	3825	RES WW 470 OHM 5% 2W BWH	R96
169	10233-78	RES MET OXIDE 15K OHM 5% 2W	R2, R22
170	10233-84	RES MET OXIDE 27K OHM 5% 2W	R21, R57, R58
171	11313-40	RES MF 2.55K OHMS 1% 1/8W	R99, R107
172	11313-35	RES MF 2.26K OHMS 1% 1/8W	R106
173	10232-60	RES MET OXIDE 1.3K OHM 5% 1W	R108
174	11306-51	RES MF 332K OHMS 1% 1/8W	R113
175	11306-93	RES MF 909K OHMS 1% 1/8W	R115
176			
177			
178	3828	RES WW .30 OHM 5% 2W BWH	R17
179	3809	RES WW .33 OHM 5% 2W BWH	R16
180	10967-41	RES WW 47K OHM 5% 5W	R56
181	3811	RES WW 10 OHM 10% 1W BW20F	R3, R4
182	3820	RES WW 120 OHM 5% 2W BWH	R32, R33, R122, R123
183	3812	RES WW 82 OHM 10% 1W BW20F	R5
184	3208	RES CF 2.2 OHM 5% 1/2W	R15
185	3903	RES WW 10 OHM 5% 5W	R25, R43, R120
186			
187			
188			
189	10519-12	RES POT 5K OHM VADJ STURN .5W CERM	R55
190	10519-10	RES POT 2K OHM VADJ STURN .5W CERM	R41, R74
191			
192	3938	RES THERMISTOR DISC 5 OHM 15% ST LD	RT1, RT2
193	13462	RES WIRE WW ZEROHM JUMPER WIRE 25A	RJ1
194			
195	3911	THERMOSTAT SNAP-ACTING AUTO RESET	S1
196	10422-11	WIRE RED 22 AWG 5"LG 1/4 X 1/4	S1, (REF)
197	7480	WIRE RED 18 AWG 1.50" (1/4 X 1/4)	T2, (REF)
198			
199	14729	TRANSFORMER XL301 VDE/RFI	T1
200	12871-02	TRANSFORMER PC MT SINGLE CORE 5T	T2
201	14923	TRANSFORMER TOROIDAL XL301	T3, T4
202			
203	12977	OPTO-RESISTOR MATCHED VDE	R70, U1
204			
205	13363	I C CMOS D PREC MONO MULTI CD4538BE	U8
206	11498	I C OPTO-ISOLATOR OP1-1264B	U3
207	14984-01	I C VOLT REF PREC 2.5V 1% MDIP	U2

Table A-15. 300W Supply Replaceable Parts (Sheet 5 of 5)
(HP 0950-1646, Boschert XL0301-5612)

ITEM NO.	COMPONENT PART NUMBER	DESCRIPTION	REFERENCE DESIGNATORS
208	14985-01	I C VOLT REF 2.5V 2% TO-92	U9
209	10505	I C LOW POWER DUAL VOLT COMP LM393N	U7
210	12219	I C OP AMP VCOMP 8 PIN MDIP LM392N	U6
211	13437	I C QUAD 2 IN SCHMITT TRIG CD4093BE	U5
212	1000	I C VOLT REG 723	U4
213	7500	TIE WRAP SMALL	S1, (REF)
214	12540	I C ADJ PREC SHUNT REG TO 92	VR1, VR2, VR3
215	7694-2	NUT KEPS CONICAL 1/4-28	CR36, CR37, (REF)
216	6009	PIN MALE BEAD .65 LG .095 DIA	E1, E2, E5
217	14957-03	JUMPER TERMINAL FEMALE	JP4
218	15696	SPRING CLIP HDWR CNDCT 12012 HTSINK	Q4, (REF)
219	11092-01	STANDOFF HEX 4-40 .250LG	P1, (REF)
220	15695	SPRING CLIP HDWR NCNDCT 12012 H/S	CR6, (REF)
221	10915	STD MTG HDW PC BD SUPPORT NYLON	
222	7501	TIE WRAP MEDIUM	
223	15451-01	STIFFENER 1.25 X 4.25" LG	C43, C44, C45, C46, C55, C56, (REF)
224	11319	RETAINING RING EXTERNAL	CR36, CR37, (REF)
225	12569	LABEL,CSA MARK	L-BRKT, (REF)
226			
227	15081-01	HANDLE .25DIA 3.0LG 1.0H 6/32THD	
228	12459	LABEL FUSE WARNING	C46, (REF)
229	14996	LABEL 115 VAC / 230 VAC	L-BRKT, (REF)
230	7740	LABEL DANGER HIGH VOLTAGE	C43, (REF)
231	7881	LABEL SERIAL NO.	T1, (REF)
232			
233	15640	SPRING CLIP HDWR (NCNDCT) TO218/220	CR32, Q7, Q8, Q9, Q15, (REF)
234	7578	SCREW P H 4-40 X 1/4	S1, (REF)
235	7511	SCREW P H 6-32 X 1/2	
236	7506	WASHER FLAT #6	
237	7588	WASHER SPLIT RING LOCK #6	
238	7577	WASHER SPLIT LOCK #4	P1, S1, L-BRKT, (REF)
239	7505	WASHER FLAT #4	P1, L-BRKT, (REF)
240	10366-01	WIRE RED 16 AWG 8" (1/4 X 1/4)	JP2, JP3
241	15633	SPRING CLIP HDWR CNDCT TO-218/220	Q5, (REF)
242	11661-01	SPACER GLASS .225 OD .067 ID .185TH	RT1, RT2, R21, R57, R58, (REF)
243	7202	0.0040RTV 108 CLEAR	L6, L7, C15, C43, C44, C45, C46, C55, C56, (REF)
244	15239	LABEL WARNING	L-BRKT, (REF)
245	15240	LABEL INFORMATION (WARNING)	L-BRKT, (REF)
246	15742	LABEL CUSTOMER I D	L-BRKT, (REF)
247	7737	0.0010LOCKTITE	
248	12891-03	PEM STUD 4-40 THREAD .625 LG	P.C. B.D., (REF)
249	7508	SCREW P H 4-40 X 3/8	L-BRKT, (REF)

A.4 HP 12154A BATTERY BACKUP MODULE

The HP 12154A battery backup module for the Micro/1000 computers is designed to provide current that will retain the current status of the system in memory if ac power is interrupted. The Hewlett-Packard part number for this card is 12154-60001. The schematic for this card is shown in Figure A-12.

A.4.1 CONFIGURATION

The battery backup module is installed in the 16-slot backplane of the Micro/1000 A-series computers. The sixth, seventh, and eighth physical slots down from the top of the left side of the card cage are dedicated to the battery backup module. There are three slots reserved to account for the height of the batteries and the space necessary for component heat sinks.

A.4.2 PHYSICAL DESCRIPTION

The module (or card) dimensions are the following:

Width: 6.75 inches (171 mm) Standard card width

Depth: 12.75 inches (324 mm) Attaches to bulkhead connector panel

Max. Component Height: 2.0 inches (51mm) for D-size battery packs

A.4.3 ELECTRICAL SPECIFICATION

The module is comprised of the battery charging circuit, enable/disable circuit, D-size battery pack, and a control circuit. When power is present, the module will charge the 9.6V Nicad (nickel/cadmium) batteries. When power fails, the 9.6V battery voltage is reduced to 5.1V through the regulator, and supplies the +5M power line to sustain memory.

The module derives its power from the 12V dc output of the system power supply. When the voltage of the batteries drops below 8.0V, the module will inhibit the +5.1V memory regulator circuit.

A.4.4 THEORY OF OPERATION

The theory of operation. The internal battery pack is charged from both 28V and 12V. The 12V backplane voltage is used to provide a heavy charge for discharged batteries. As the batteries charge to about 11.3V, the 28V supply is used as a trickle charger. The input selector is essentially three diodes that select the highest input voltage. Note the 28V supply would be the normal input, until a power fail when the batteries would become the input. This is important because the output of this circuit, +5VM, is produced by

the circuitry on this card. Thus to determine if the battery or card is at fault, the battery pack can be disabled, by turning of the battery enable switch on the front panel, and if the +5VM is present the battery pack is probably at fault (if the 28V and 5V voltages are present).

The DC-to-DC converter, coupled with the control circuit, make this a switching power supply. The control circuit switches the DC input voltage to maintain a pulse width with a constant amount of power. Thus, a 28V input will have a tall narrow pulse, a 8V input will have a low wide pulse. The pulses feed an inductor that creates a constant voltage output for a capacitor-resistor integrator that produces the +5VM. This supply will stop if the battery voltage drops below 8V (to protect the ni-cad batteries) and will shut down if an over current condition exists. The 2.5V reference has a +/- accuracy of 0.5%. The feedback circuit drives the control circuit to regulate the pulse width of the DC-to-DC converter. Overall voltage regulation is 2%.

The battery charger consists of four parts: Diode CR2 is connected to the +12V supply and to a 10 ohm resistor, R2. R2 is connected to the positive side of the battery. Diode CR1 is similarly connected to +28V and 100 ohm resistor R1 which is also connected to the positive battery connection. Diode CR2 conducts until the battery voltage approaches 11.3V. CR2 becomes reverse biased and CR1 now trickle charges the battery.

On the front of the card are two LEDs, one green and one red. The red LED indicates that an overcurrent of overvoltage condition has occurred and the battery backup card is shutdown. This could be caused by a short in the +5VM line that goes to memory. Either a memory card, a processor card or the backplane could be faulty, as well as the battery backup card itself. To reset after isolating and repairing the problem, turn off the battery enable switch, turn off the power, then turn both back on.

The green LED indicates the +5VM is within specifications. If both red and green LEDs are on, the battery backup card is faulty. As long as the green LED is on, the memory is being sustained.

The computer must be on and the battery enable switch turned on to charge the batteries. The battery enable switch should be turned off when removing/inserting assemblies to avoid arcing and transients.

A.4.5 REPLACEABLE PARTS

Replaceable parts for the HP 12154A are listed in Table A-16 and a parts location diagram for it is shown in Figure A-13.

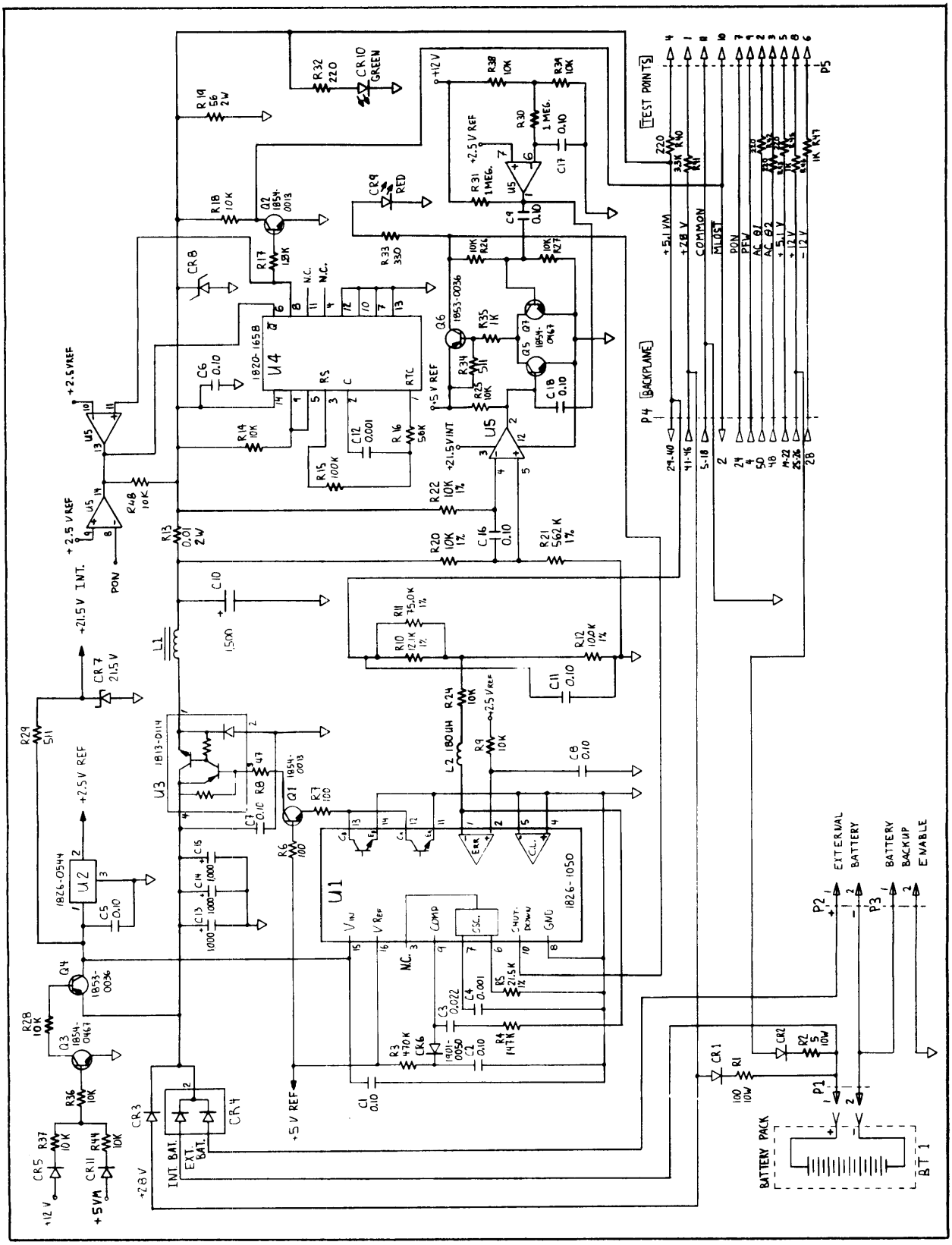


Figure A-12. HP 12154A Battery Backup Module Schematic

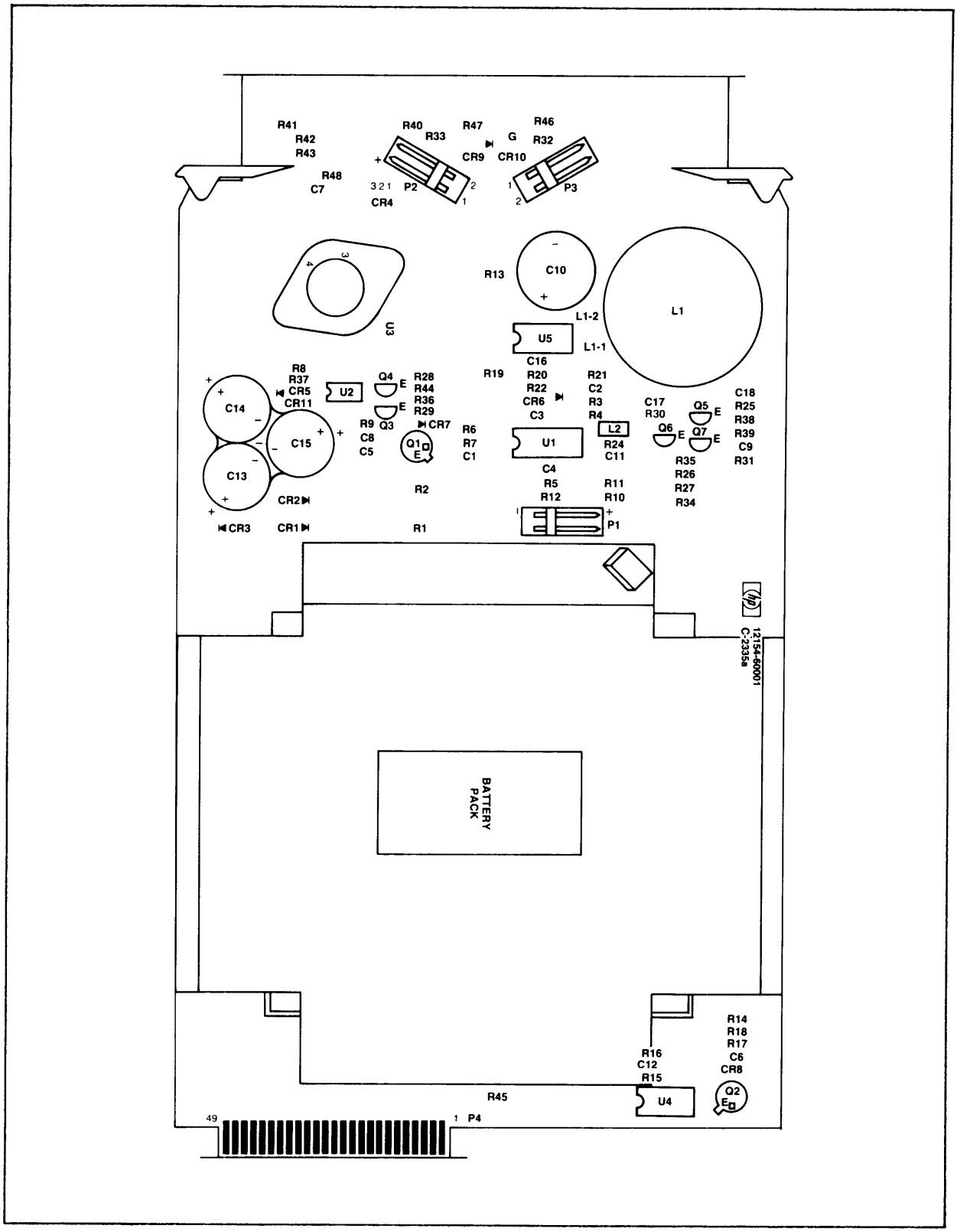


Figure A-13. HP 12154A Replaceable Parts Location Diagram

Table A-16. HP 12154A Replaceable Parts (Sheet 1 of 2)

Reference Designation	HP Part Number	C D	Qty	Description	Mfr Code	Mfr Part Number
	12154-60001	3	1	PCA-BACKUP	28480	12154-60001
B1	1420-0321	1	1	BATTERY ASSEMBLY 9.6V NOM; RECHARGEABLE	28480	1420-0321
C1	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C2	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C3	0160-4833	5		CAPACITOR-FXD .022UF +-10% 100VDC CER	28480	0160-4833
C4	0160-4847	1		CAPACITOR-FXD 1000PF +-10% 100VDC CER	28480	0160-4847
C5	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C6	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C7	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C8	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C9	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C10	0180-2997	0	1	CAPACITOR-FXD 1500UF+100-10% 25VDC AL	56289	674D158H025HJ5A
C11	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C12	0160-4847	1		CAPACITOR-FXD 1000PF +-10% 100VDC CER	28480	0160-4847
C13	0180-3019	9	3	CAPACITOR-FXD 1000UF+50-10% 50VDC AL	28480	0180-3019
C14	0180-3019	9		CAPACITOR-FXD 1000UF+50-10% 50VDC AL	28480	0180-3019
C15	0180-3019	9		CAPACITOR-FXD 1000UF+50-10% 50VDC AL	28480	0180-3019
C16	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C17	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
C18	0160-4835	7		CAPACITOR-FXD .1UF +-10% 50VDC CER	28480	0160-4835
CR1	1901-0673	6	3	DIODE-PWR RECT 100V 5A SUS	03508	A15A
CR2	1901-0673	6		DIODE-PWR RECT 100V 5A SUS	03508	A15A
CR3	1901-0673	6		DIODE-PWR RECT 100V 5A SUS	03508	A15A
CR4	1906-0265	2	1	DIODE-RECT.	28480	1906-0265
CR5	1901-0050	3	4	DIODE-SWITCHING 80V 200MA 2NS DO-35	28480	1901-0050
CR6	1901-0050	3		DIODE-SWITCHING 80V 200MA 2NS DO-35	28480	1901-0050
CR7	1902-3245	6	1	DIODE-ZNR 21.5V 5% DO-35 PD=.4W	28480	1902-3245
CR8	1902-0939	9	1	DIODE-ZNR 5V PD=5W TC=+.06% IR=300UA	11961	1N5908
CR9	1990-0486	6	1	LED-LAMP LUM-INT=1MCD IF=20MA-MAX BVR=5V	28480	5082-4684
CR10	1990-0485	5	1	LED-LAMP LUM-INT=800UCD IF=30MA-MAX	28480	5082-4984
CR11	1901-0460	9	2	DIODE-STABISTOR 30V 150MA DO-7	28480	1901-0460
L1	9140-0811	8	1	INDUCTOR 356UH 10% 1.7DX1.2LG	28480	9140-0811
L2	9100-2279	2		INDUCTOR RF-CH-MLD 180UH 10% .105DX.26LG	28480	9100-2279
Q1	1854-0013	7	2	TRANSISTOR NPN 2N2218A SI TO-5 PD=800MW	04713	2N2218A
Q2	1854-0013	7		TRANSISTOR NPN 2N2218A SI TO-5 PD=800MW	04713	2N2218A
Q3	1854-0467	5	3	TRANSISTOR NPN 2N4401 SI TO-92 PD=310MW	03508	2N4401
Q4	1853-0036	2	2	TRANSISTOR PNP SI PD=310MW FT=250MHZ	28480	1853-0036
Q5	1854-0467	5		TRANSISTOR NPN 2N4401 SI TO-92 PD=310MW	03508	2N4401
Q6	1853-0036	2		TRANSISTOR PNP SI PD=310MW FT=250MHZ	28480	1853-0036
Q7	1854-0467	5		TRANSISTOR NPN 2N4401 SI TO-92 PD=310MW	03508	2N4401
R1	0811-3644	5	1	RESISTOR 100 10% 10W PW TC=0+-300	28480	0811-3644
R2	0811-3656	9	1	RESISTOR 5 10% 10W PW TC=0+-300	28480	0811-3656
R3	0683-4745	6	2	RESISTOR 470K 5% .25W FC TC=-800/+900	01121	CB4745
R4	0698-3452	1	2	RESISTOR 147K 1% .125W F TC=0+-100	24546	C4-1/8-T0-1473-F
R5	0757-0199	3		RESISTOR 21.5K 1% .125W F TC=0+-100	24546	C4-1/8-T0-2152-F
R6	0683-1015	7		RESISTOR 100 5% .25W FC TC=-400/+500	01121	CB1015
R7	0683-1015	7		RESISTOR 100 5% .25W FC TC=-400/+500	01121	CB1015
R8	0683-4705	8		RESISTOR 47 5% .25W FC TC=-400/+500	01121	CB4705
R9	0683-1035	1	26	RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R10	0757-0444	1		RESISTOR 12.1K 1% .125W F TC=0+-100	24546	C4-1/8-T0-1212-F
R11	0757-0462	3	2	RESISTOR 75K 1% .125W F TC=0+-100	24546	C4-1/8-T0-7502-F
R12	0757-0442	9	8	RESISTOR 10K 1% .125W F TC=0+-100	24546	C4-1/8-T0-1002-F
R13	0811-3511	5	1	RESISTOR .01 1% 2W PWW TC=0+-150	28480	0811-3511
R14	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R15	0683-1045	3		RESISTOR 100K 5% .25W FC TC=-400/+800	01121	CB1045
R16	0683-5635	5		RESISTOR 56K 5% .25W FC TC=-400/+800	01121	CB5635
R17	0683-1825	7		RESISTOR 1.8K 5% .25W FC TC=-400/+700	01121	CB1825
R18	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R19	0764-0013	5	1	RESISTOR 56 5% 2W MO TC=0+-200	28480	0764-0013
R20	0757-0442	9		RESISTOR 10K 1% .125W F TC=0+-100	24546	C4-1/8-T0-1002-F
R21	0698-8824	1		RESISTOR 562K 1% .125W F TC=0+-100	28480	0698-8824
R22	0757-0442	9		RESISTOR 10K 1% .125W F TC=0+-100	24546	C4-1/8-T0-1002-F
R24	0757-0442	9		RESISTOR 10K 1% .125W F TC=0+-100	24546	C4-1/8-T0-1002-F
R25	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R26	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R27	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R28	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R29	0757-0416	7	4	RESISTOR 511 1% .125W F TC=0+-100	24546	C4-1/8-T0-511R-F
R30	0683-1055	5		RESISTOR 1M 5% .25W FC TC=-800/+900	01121	CB1055
R31	0683-1055	5		RESISTOR 1M 5% .25W FC TC=-800/+900	01121	CB1055

Table A-16. HP 12154A Replaceable Parts (Sheet 2 of 2)

Reference Designation	HP Part Number	C D	Qty	Description	Mfr Code	Mfr Part Number
R32	0683-2215	1	10	RESISTOR 220 5% .25W FC TC=-400/+600	01121	CB2215
R33	0683-3315	4		RESISTOR 330 5% .25W FC TC=-400/+600	01121	CB3315
R34	0757-0416	7		RESISTOR 511 1% .125W F TC=0+-100	24546	C4-1/8-T0-511R-F
R35	0683-1025	9	6	RESISTOR 1K 5% .25W FC TC=-400/+600	01121	CB1025
R36	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R37	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R38	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R39	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R40	0683-2215	1		RESISTOR 220 5% .25W FC TC=-400/+600	01121	CB2215
R41	0683-3325	6	2	RESISTOR 3.3K 5% .25W FC TC=-400/+700	01121	CB3325
R42	0683-2215	1		RESISTOR 220 5% .25W FC TC=-400/+600	01121	CB2215
R43	0683-2215	1		RESISTOR 220 5% .25W FC TC=-400/+600	01121	CB2215
R44	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
R45	0683-2215	1		RESISTOR 220 5% .25W FC TC=-400/+600	01121	CB2215
R46	0683-1025	9		RESISTOR 1K 5% .25W FC TC=-400/+600	01121	CB1025
R47	0683-1025	9		RESISTOR 1K 5% .25W FC TC=-400/+600	01121	CB1025
R48	0683-1035	1		RESISTOR 10K 5% .25W FC TC=-400/+700	01121	CB1035
U1	1826-1050	5	1	IC V RGLTR-SWG 4.9/5.1V 16-DIP-P PKG	28480	1826-1050
U2	1826-0544	0	1	V REF 8-DIP-C	04713	MC1403U
U3	1813-0114	3	1	IC V RGLTR TO-3	12969	PIC645
U4	1820-3516	0	1	IC TIMER CMOS	28480	1820-3516
U5	1826-0138	8	1	IC COMPARATOR GP QUAD 14-DIP-P PKG	01295	LM339N

The parts manufacturer's names and addresses are listed in the Manufacturer's Code List below.

Manufacturer's Code List

MFR NO.	MANUFACTURER NAME	ADDRESS	ZIP CODE
01121	Allen-Bradley Co	Milwaukee, WI	53204
01295	Texas Instr Inc Semicond Cmpnt Div	Dallas, TX	75222
03508	GE Co Semiconductor Prod Dept	Auburn, NY	13201
04713	Motorola Semiconductor Products	Phoenix, AZ	85008
11236	CTS of Berne Inc	Berne, IN	46711
11961	Semicon Inc	Burlington, MA	01803
12969	Unitrode Corp	Watertown, MA	02172
24546	Corning Glass Works (Bradford)	Bradford, PA	16701
28480	Hewlett-Packard Co. Corporate Hq	Palo Alto, CA	94304
31585	RCA Corp Solid State Div	Somerville, NJ	08876
56289	Sprague Electric Co.	North Adams, MA	01247
75915	Littlefuse Inc	Des Plaines, IL	60016

A.5 HP 12159A 25 kHz POWER MODULE

The HP 12159A is a module that takes the 25 kHz square-wave power from the 300W power supply and provides 25 kHz sine-wave power to the 16-slot backplane of a Micro/1000 computer for distribution to certain I/O cards having on-card power supplies. The module plugs into slot 8 of the backplane, and does not need forced air flow for cooling.

The Hewlett-Packard part number for this module is 12159-60001.

A.5.1 HP 12159A SPECIFICATIONS

The HP 12159A electrical specifications are provided in Table A-17, and the connector pin definitions are listed in Table A-18.

Table A-17. HP 12159A 25-kHz Module Electrical Specifications

Input Specifications:

	Min	Nominal	Max	
Phase to Phase	11.28	12.00	12.72	Volts rms
RMS Current (max)	0.00	-	3.20	Amps
Frequency	24	25	32	kHz
Input Power	-	-	36	Watts

Output Specifications:

Maximum output power	30 Watts
Maximum distortion	10% total harmonic distortion
Maximum dynamic load change	10% of load over 10 microseconds

Output Stress Conditions:

The module shall recover, with no permanent damage from a shorted regulated output (it may be necessary to replace the fuse to resume normal operation)

Output Loading:

The load on the 25 kHz module can be applied from phase-to-phase (39 Vrms) or from phase-to-common (19.5 Vrms). Up to one half of the maximum rated power can be drawn from each phase (phase-to-common) or up to all of the rated power can be drawn phase-to-phase as long as the total power does not exceed the maximum rated power. The phase to common load need not be balanced between between the two phases.

Regulated Output Specification:

Nominal Voltage	39	Volts rms
Split Phase	19.5	Volts rms
Maximum Current	0.84	Amps

Regulation:

0.0 to 0.02 Amps	+10%	-12%
0.02 to 0.84 Amps	+8%	-8%

Table A-18. HP 12159A Electrical Connector (P1) Pin Definitions

PIN	SIGNAL NAME
11-14	25 kHz1
15-18	25 kHz2
19-22	GND
23-26	AC02 (phase 2)
27-30	AC01 (phase 1)

A.5.2 HP 12159A THEORY OF OPERATION

For this theory of operation, refer to the schematic shown in Figure A-14. The input transformer T1 steps the input 24 Volt peak-to-peak square wave up to 114 Volts peak-to-peak across 3-6. The winding 5-6 regulates the output as described below.

L1, L2, C1, and C2 are the main harmonic filters of the square wave. The regulator coil L3 also contributes to filtering.

Components R1, R2, C3, and C4 attenuate the noise generated by the switching diodes CR1 and CR2.

The regulator limits the amplitude of the output sine wave to less than 59 Volts and imposes no minimum value. The output of CR1, CR2, and C5 is the peak of the output sine wave. If this voltage exceeds 26 volts (zener diode CR3 voltage) plus the B-E drops across Q1 and Q2, then a current flows into the base of Q1, causing current flow through CR4 and Q2. This current adds to the main circuit current through L3, causing a voltage drop across L3. This voltage drop is a strong function of excessive output voltage which has the effect of providing regulation.

The regulator clamps the output to be less than or equal to the sum of the zener voltage plus the B-E drops of the transistors.

A.5.3 REPLACEABLE PARTS

Replaceable parts for the 12159A are listed in Table A-19 and the names and addresses of the parts manufacturers are listed in the Manufacturer's Codes List below. The parts locations are shown in Figure A-15.

Manufacturer's Code List

MFR NO.	MANUFACTURER NAME	ADDRESS	ZIP CODE
00000	Any Satisfactory Supplier		
01121	Allen-Bradley Co	Milwaukee, WI	53204
01295	Texas Instr Inc Semicond Cmpnt Div	Dallas, TX	75222
03888	K D I Pyrofilm Corp	Whippany, NJ	07981
04713	Motorola Semiconductor Products	Phoenix, AZ	85008
07263	Fairchild Semiconductor Div	Mt. View, CA	94042
07910	Teledyne Semiconductor	Hawthorne, CA	90250
11236	CTS of Berne Inc	Berne, IN	46711
11961	Semicon Inc	Burlington, MA	01803
14936	General Instr Corp Semicon Prod Gp	Hicksville, NY	11802
19701	Mepco/Electra Corp	Mineral Wells, TX	76067
24546	Corning Glass Works (Bradford)	Bradford, PA	16701
27014	National Semiconductor Corp	Santa Clara, CA	95051
28480	Hewlett-Packard Co. Corporate Hq	Palo Alto, CA	94304
32293	Intersil Inc	Cupertino, CA	95014
34335	Advanced Micro Devices Inc	Sunnyvale, CA	94086
34649	Intel Corp	Mt. View, CA	95051
50088	Mostek Corp	Carrollton, TX	75006
50364	Monolithic Memories Inc	Sunnyvale, CA	94086
56289	Sprague Electric Co	North Adams, MA	01247

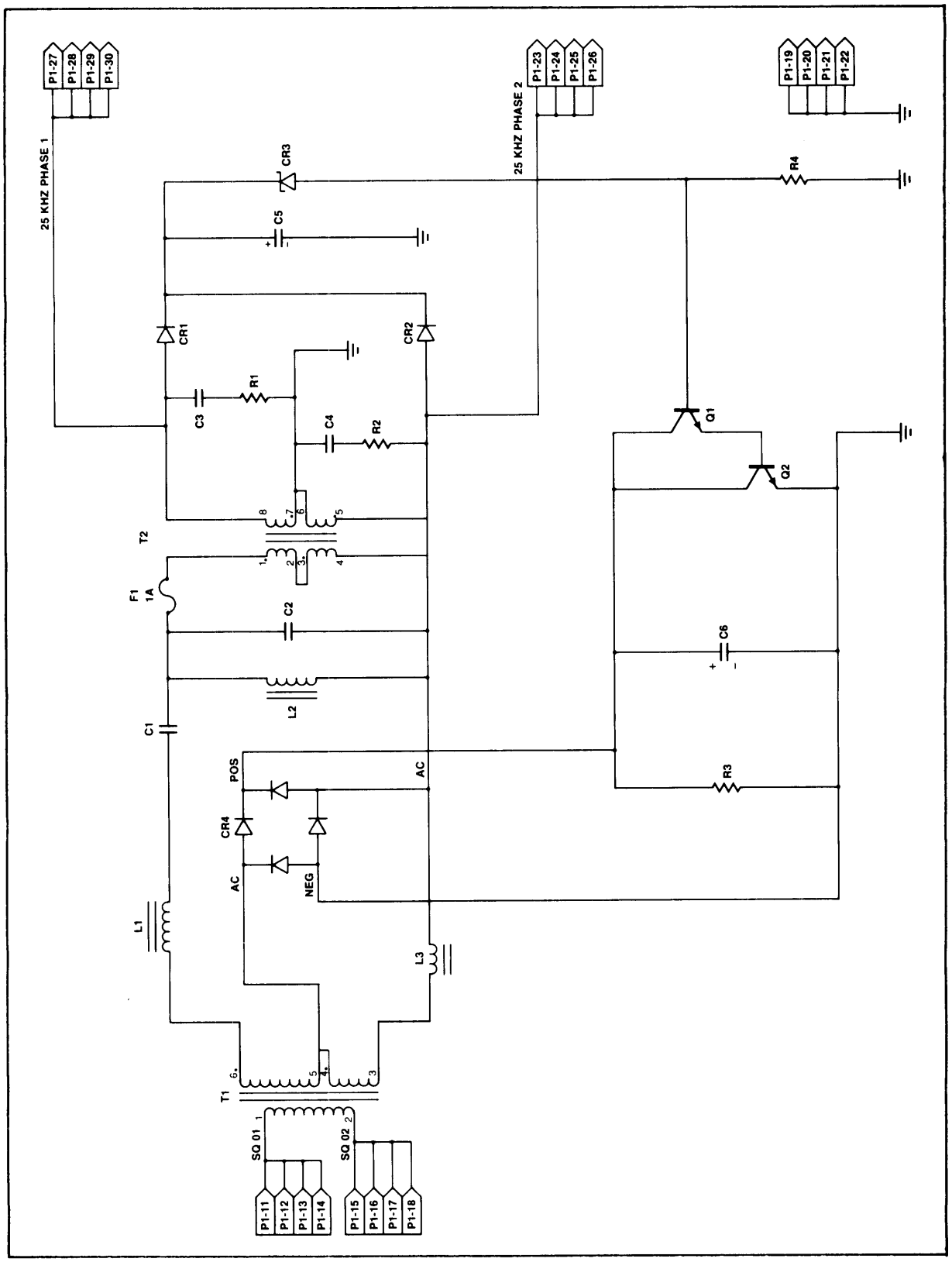


Figure A-14. HP 12159A 25 kHz Power Module Schematic

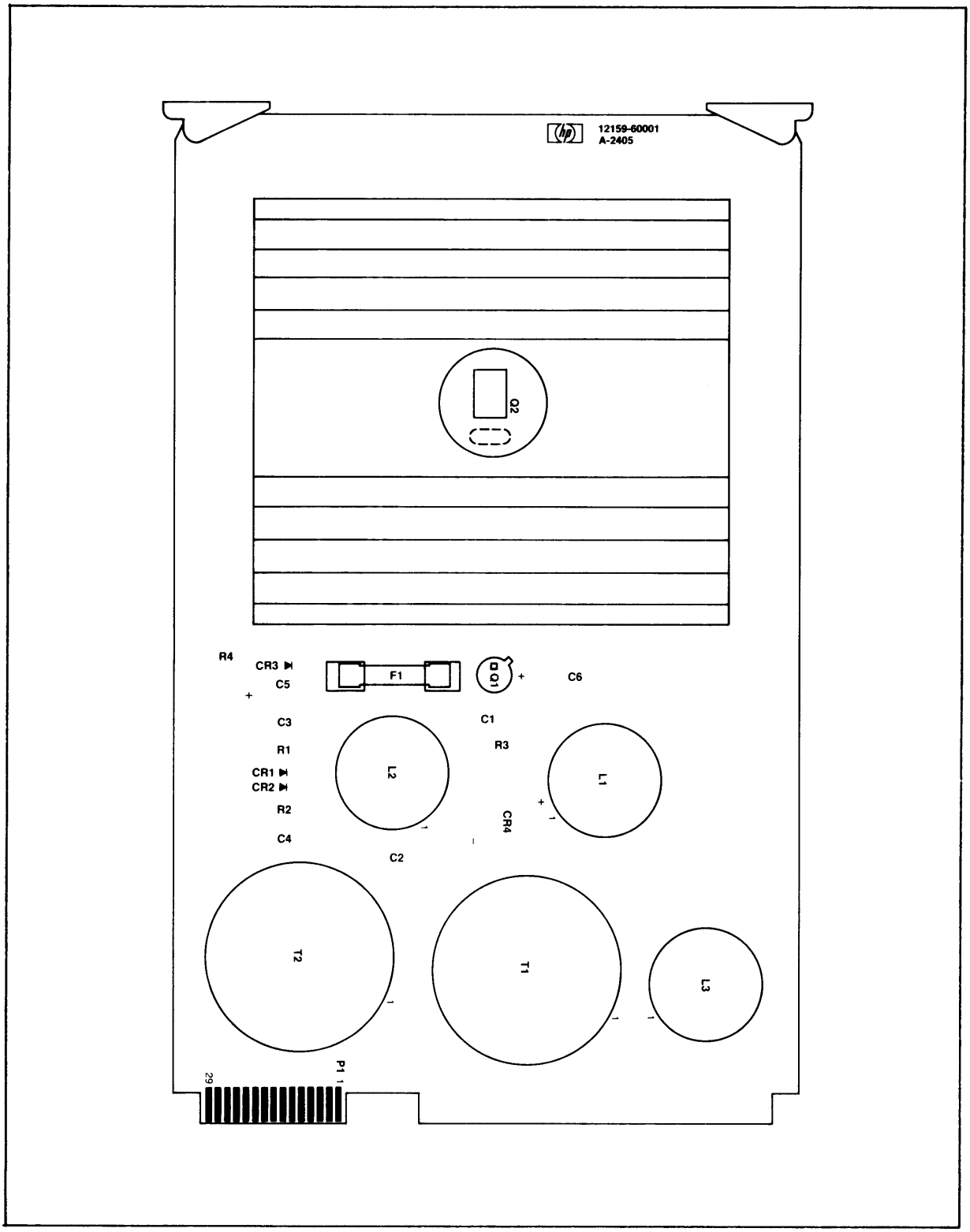


Figure A-15. HP 12159A Replaceable Parts Locations

Table A-19. HP 12159A Replaceable Parts

Reference Designation	HP Part Number	CD	Qty	Description	Mfr Code	Mfr Part Number
	12159-60001	8	1	PCA-25KHZ	28480	12159-60001
C1	0160-6040	0	2	CAP 0.1 UF	28480	0160-6040
C2	0160-6040	0		CAP 0.1 UF	28480	0160-6040
C3	0160-0161	4	2	CAPACITOR-FXD .01UF +-10% 200VDC POLYE	28480	0160-0161
C4	0160-0161	4		CAPACITOR-FXD .01UF +-10% 200VDC POLYE	28480	0160-0161
C5	0180-0269	5	1	CAPACITOR-FXD 1UF+50-10% 150VDC AL	56289	30D105G150BA2
C6	0180-0141	2	1	CAPACITOR-FXD 50UF+75-10% 50VDC AL	56289	30D506G050DD2
CR1	1901-0096	7	2	DIODE-SWITCHING 120V 50MA 100NS	28480	1901-0096
CR2	1901-0096	7		DIODE-SWITCHING 120V 50MA 100NS	28480	1901-0096
CR3	1902-3269	4	1	DIODE-ZNR 26.1V 2% DO-35 PD=.4W	28480	1902-3269
CR4	1906-0077	4	1	DIODE-FW BRDG 400V 5A	28480	1906-0077
F1	2110-0001	8	1	FUSE 1A 250V NTD 1.25X.25 UL	75915	312001
L1	9140-0863	0	1	IND-FXD 240UH	28480	9140-0863
L2	9140-0861	8	1	IND-FXD 335 UH	28480	9140-0861
L3	9140-0862	9	1	IND-FXD 95UH	28480	9140-0862
Q1	1854-0079	5	1	TRANSISTOR NPN 2N3439 SI TO-5 PD=1W	3L585	2N3439
Q2	1854-0727	0	1	TRANSISTOR NPN 2N6474 SI TO-220AB PD=40W	3L585	2N6474
R1	0698-3620	5	2	RESISTOR 100 5% 2W MO TC=0+-200	28480	0698-3620
R2	0698-3620	5		RESISTOR 100 5% 2W MO TC=0+-200	28480	0698-3620
R3	0683-2735	0	1	RESISTOR 27K 5% .25W FC TC=-400/+800	01121	CB2735
R4	0683-2215	1	1	RESISTOR 220 5% .25W FC TC=-400/+600	01121	CB2215
T1	9140-0859	4	1	XFMR	28480	9140-0859
T2	9140-0860	7	1	XFMR	28480	9140-0860

A.6 25 kHz BACKPLANE POWER APPLICATIONS

25 kHz backplane power can be used when designing special interfaces on the 12010A Breadboard Interface to provide ac input power for compact, lightweight on-interface dc power supplies to meet any of the following requirements.

1. Provision of dc voltages in addition to those supplied by the power supply.

2. Provision of dc supplies whose analog grounds are isolated from the computer ground.

3. Provision of multichannel isolated power to digital communication circuits to eliminate ground noise paths and maximize the reliability of serial data transfers.

4. Low voltage, high current power for supplying large arrays of integrated circuits.

A.6.1 NON-ISOLATED POWER SUPPLY

A.6.1.1 Purpose and Basic Design

Where additional +7.5V to +12V dc at up to 1 amp is needed for interface circuits, the 25 kHz backplane power can be used to provide a non-isolated positive regulated power supply as shown in Figure A-16. The 19.5V rms potential on either side of common provides at least +15.4V dc after rectification and filtering. An adjustable, off-the-shelf, three-terminal integrated circuit voltage regulator (National Semiconductor Series LM11 7 or equivalent) can be used to set the regulated output voltage within the range of +7.5V to +12V dc. The regulated voltage output is dependent upon the values of resistors R2 and R3. A negative output voltage supply similar to the positive supply shown in Figure A-16 can be made by reversing the polarities of the rectifiers and using a negative adjustable regulator, (National Semiconductor Series LM137 or equivalent).

A.6.1.2 Preserving Purity of Input Sine Wave

To maintain the purity of the input 25 kHz sine wave, near 180 degree conduction should be provided in the rectification process, which necessitates the use of a choke input filter. This filter also limits the surge current at turn-on if the requirements for Lmin are met. The equation for Lmin with a 25% safety factor is given by:

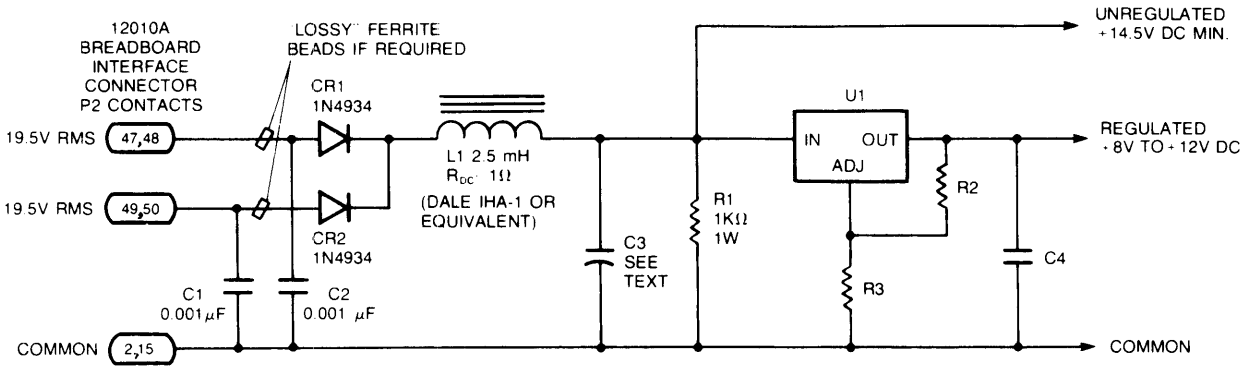
Lmin (in henries) = (K/fs) x Rl

where: Fs = 25 kHz
Rl = Minimum load resistance
K = 0.06 for full wave rectifiers

This implies the need for a minimum load. If the circuits to be powered allow the load current to go to zero, a preloading bleeder resistor is required. The final value of Lmin would then be determined by the allowed power loss (dissipation) of the preloading resistor. When the Lmin requirement is met, the surge current will be acceptable and sine wave distortion will be minimized.

A.6.1.3 Rectifier Selection

Rectifiers used with 25 kHz input power must be of the fast recovery type with less than 200 nanosecond recovery time. Allowing for possible transients from leakage instances, overshoot, and MTBF derating, the rectifiers should also have 100V peak inverse voltage rating.



NOTES: U1 is a National Semiconductor type LM117 Series or equivalent adjustable regulator.
Values of C4, R2, and R3 should be selected in accordance with instructions in U1 manufacturer's data sheet.

Figure A-16. On-interface Regulated Power Supply

A.6.1.4 Input Noise Reduction

During rectifier recovery, the removal of stored charge in the rectifiers will appear as spikes on the rectifier inputs. These spikes should be suppressed to keep them from travelling along the 25 kHz ac input lines in the backplane. Small 0.001 to 0.1 microfarad ceramic capacitors (C1 and C2 in Figure A-16) will usually damp out these spikes, with the required capacitor value dependent upon the magnitude of stored charge being removed. If underdamped ringing is present because of leakage inductance, small ferrite beads, tubes, or toroids can be threaded onto the rectifier leads to provide a "lossy" inductive reactance at high frequencies to effectively dissipate undesirable recovery currents.

A.6.1.5 Input Filtering

The value of C3 is determined by the amount of ripple voltage that can be tolerated at the input of integrated circuit regulator U1. The Vin-Vout differential of 3 volts must be met for any chosen output voltage as noted in Reference 2. The Ripple factor r for a full-wave rectifier circuit is given by:

$$r = (0.83/(L1 \times C1) \times 5.76 \times 10^{-6}$$

The case size and construction of capacitor C3 must be capable of conducting the ripple current without excessive dissipation. Ripple current will be at 2 fs and will be sinusoidal when Lmin requirements are met. The rms ripple current in amps is given by:

$$1r = VRMS/(4\pi \times fs \times L1)$$

Where: VRMS is the input voltage phase to common

fs = 25 kHz
L1> = Lmin

The minimum inductive value of L1 must be present with the dc current flowing through it over the complete load current range. This requires an inductor with gaps in the magnetic circuit, either fixed or distributed, such as in powdered iron cores, or solenoid-wound inductors over ferrite rods (available from Reference 9).

A.6.1.6 Regulator Dissipation

Since the regulator is a linear series pass type, the difference between the voltage developed across C3 at the regulator input and the desired output at the load current must be dissipated in the regulator. This dissipation is given by:

$$Pdiss = (Vin-Vout) \times (IL + Vin\ Iq)$$

Where: Iq = the quiescent current of the regulator.

Case to junction thermal resistances are given in the regulator manufacturer's data sheet. The dominant thermal resistance will be the case to air stream, which is usually available on heat sink manufacturer's data as a function of air velocity. You can assume a minimum 200 ft/min flow across the board with a maximum air temperature on the exit side of 66 C under worst case conditions. For low power on-card dc supplies, the copper foil on the printed circuit board can be used as a heat sink. However, the suitability of this arrangement should be checked carefully with thermocouples to confirm that the temperature rise of the regulator is not excessive.

A.6.2 ISOLATED OR FLOATING DC POWER

A major advantage of the 25 kHz backplane power is its ease of use for isolated power supplies that can have separate analog grounds, thereby reducing the effects of ground-conducted noise as discussed in References 3 and 4. Isolation is provided by an on-interface transformer, as shown in Figure A-17. The use of 25 kHz ac input makes it possible for the isolation transformer to be very small and inexpensive. Toroidal printed circuit mounting types or "P" core (Reference 7) shielded printed circuit mounting types generally offer the best price-performance combination. However, small E-E types can also be used at lower cost with some sacrifice in electromagnetic and electrostatic shielding. High permeability ferrite materials having low losses at 25 kHz are readily available with matching bobbins and mounting hardware from References 6 through 10.

Primary-to-secondary isolation of both dc and high frequency can be somewhat complex. References 3 and 4 describe single and double shielded transformers. It is possible to achieve high isolation with small ferrite cores and proper inter-winding shield design. Simple copper foil inter-winding shields are relatively inexpensive and are effective in decreasing primary-to-secondary electrostatic coupling at frequencies from 100 Hz to about 100 kHz. For higher frequencies, "link" coupling of two cores or other techniques may be required (Reference 3, p 117).

The ground isolation provided by the multi-channel +10V 30 MA power supply circuits depicted in Figure A-17 eliminates errors caused by ground-induced noise. In analog voltage measurement applications, power supply isolation minimizes common mode noise, improving measurement accuracy. With respect to digital data transmission uses, power supply isolation allows data terminals to operate at greater distances from the local system with fewer data errors than would otherwise be possible. When the power supply is not isolated, noise in the 50/60 Hz mains power distribution and grounding system supplying the computer can cause current noise loops that degrade signal integrity.

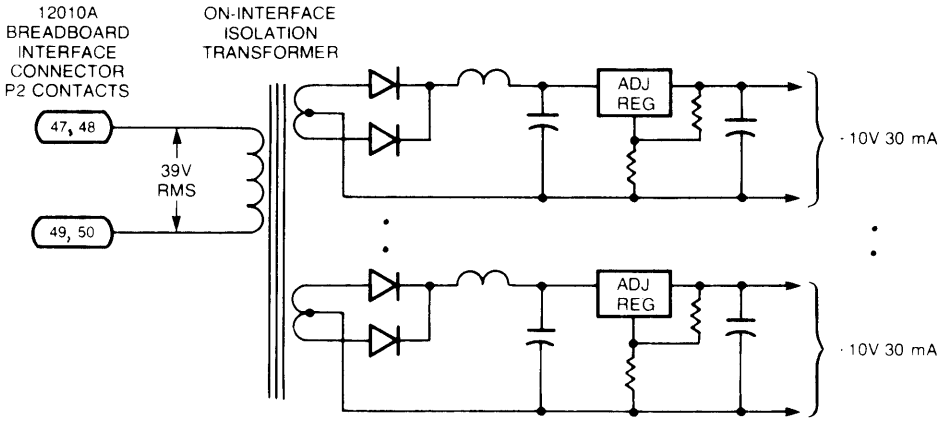


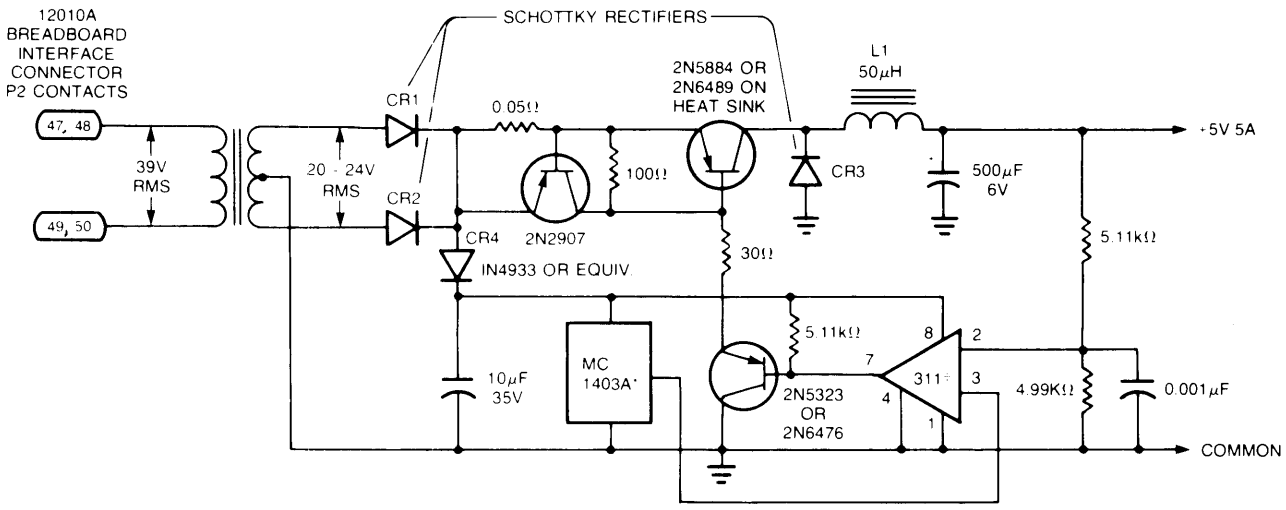
Figure A-17. Multiple, Isolated, On-interface Power Supplies

A.6.3 LOW-VOLTAGE, HIGH-CURRENT POWER SUPPLY

Heat dissipation is often the main factor limiting the current output of on-interface power supplies. This is particularly true for lower voltage, high current supplies, such as required for many digital integrated circuit families. For example, at the +5V used for TTL families of integrated logic circuits, even the dissipation of the rectifiers can be a significant 40% to 20% of total power, because of the inherent 0.7V to 1.0V forward drop across silicon rectifiers, and heat sinking may be required at 3-5 Amp currents. Use of hot carrier or Schottky junction rectifiers, which have a lower forward drop presenting a power loss of only 4%-5% of the total power output, have peak inverse voltage ratings that are suitable for lower voltage power supplies and may not require heat sinks because of their lower power dissipation.

At low output voltages, the 2-3 volt drop required across most three-terminal adjustable integrated circuit series regulators for proper regulation can account for 40%-60% of the total power output, which is lost in the regulator and must be dissipated. Regulator heat sinking becomes difficult for even 1-3 Amp current outputs and impossible for the higher current levels that larger three-terminal regulators are able to pass. Because of these efficiency and dissipation problems, a more efficient circuit approach has evolved, as shown in Figure A-18.

The circuit of Figure A-18 uses a driven switching regulator for more efficient delivery of low voltage, high current output. This circuit regulates on the basis of the conduction angle of the pulsating rectified, unfiltered dc from the on-interface Schottky rectifiers. The result is efficiencies of 70%-85% with 1 Amp to 5 Amp loads. The duty cycle control is uniform over the half sine wave and the instantaneous energy is low at the switching transitions, which minimizes waveform distortion and RFI emmision. Because the regulator operates on the incoming frequency as a driven circuit, it also eliminates the generation of other frequencies that would be a problem if an on-interface switching regulator integrated circuit were used. The circuit eliminates sum and difference noise frequencies and a host of non-repetitive noise problems, while optimizing efficiency.



CR1, CR2, and CR3 are International Rectifier 80SQ10 5A Schottky rectifiers
**Motorola MC1403A or equivalent 2.5V low TC reference source*
†National Semiconductor LM 311 or equivalent Comparator
L1 is a Dale type IH5 or equivalent solenoid choke coil

Figure A-18. On-interface, High Current Switching Power Supply

References:

1. Reference Data for Radio Engineers, Fifth Edition, Howard W. Sams & Co., Inc., 1974; Chapter 13, pp 28-30.

2. National Semiconductor Linear, Data Book, 1978, Section I, pp 15-22 and 50-54.

3. Morrison, Ralph, "Grounding and Shielding Techniques in Instrumentation", Second Edition, Wiley Publications, Inc., 1977.

4. Ott, Henry, "Noise Reduction Techniques in Electronic Systems", Wiley Publications, Inc., 1976.

5. Fairchild "Voltage Regulator Handbook" or "Hybrid Data Book", available from Fairchild Semiconductor.

6. Ferroxcube "Linear Ferrite Materials and Components".

7. TDK Data Book, Ferrite Cores - 2 DLE 88-002A.

8. Siemens Data Book, "Soft Magnetic Siferit", 1975.

9. Fair-Rite Materials Data Book (Rods).

10. Micrometals "Shielded Coil Forms".

11. White, Donald, "EMI Control Methodology and Procedures", Don White Consultants, 1978.

REV	DESCRIPTION	ECO	DATE	APPROVED
X	SEE SH 2			

BD. NO.1 (BOTTOM)

REFERENCE DESIGNATIONS 3

LAST USED	NOT USED
A3	A1
C146	C115
CR128	
E36	E1,2
F3	
JP6	JP1, JP2
L103	
Q105	
R156	
S100	
T3	
U102	

NOTES ARE ON SH 2.

ITEM	QTY	PART NUMBER	DESCRIPTION	REF DESIGNATION

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES:
DECIMALS (.XX) ± FRACTIONS ±
DECIMALS (.XXX) ± ANGLES ±

NEXT ASSY	MOD NO.
45-13606-01	XL40J-5411R

DO NOT SCALE DRAWING

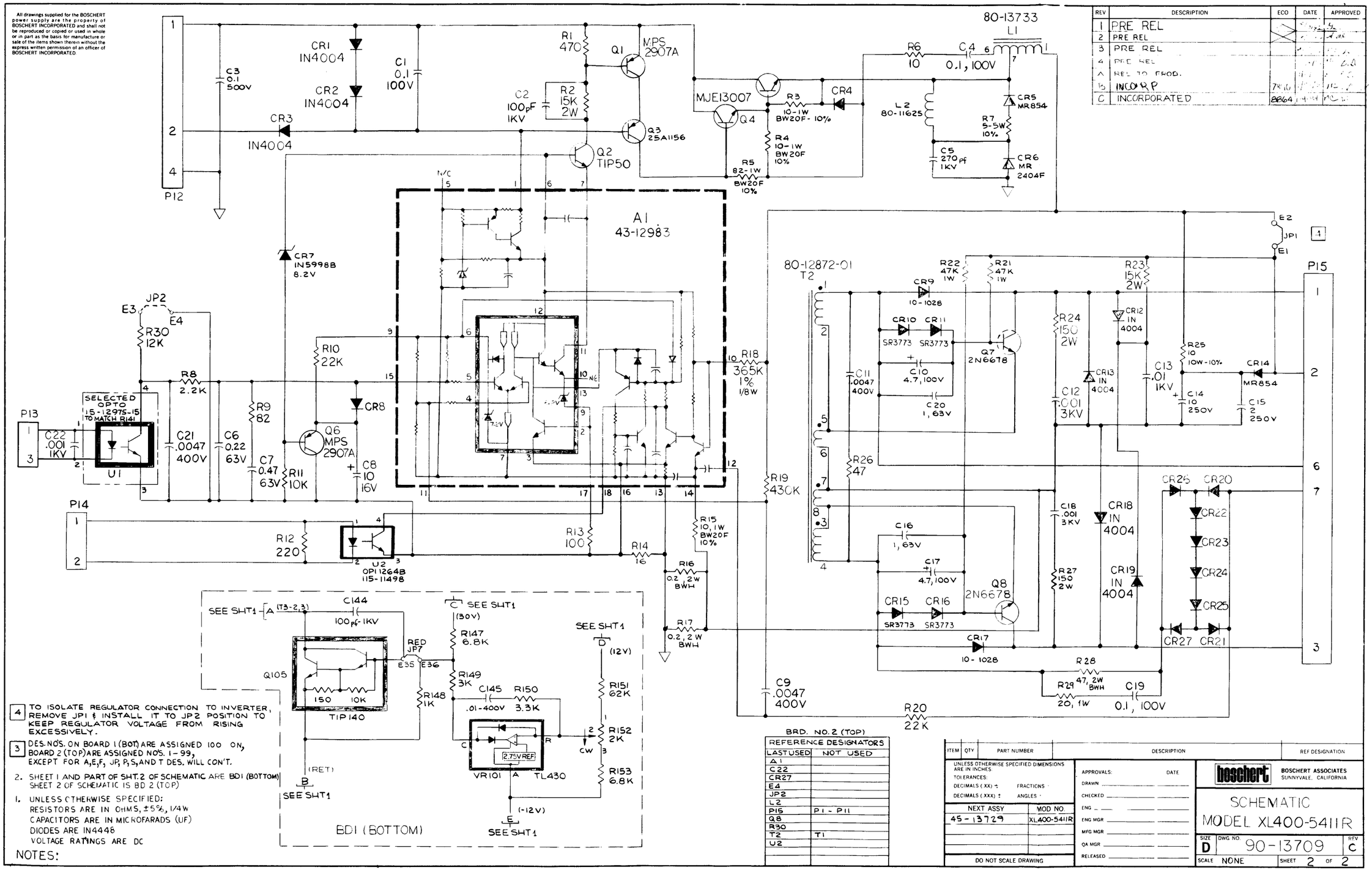
BOSCHERT ASSOCIATES
SUNNYVALE, CALIFORNIA

SCHEMATIC,
MODEL: XL400-5411R

SIZE	DWG NO.	REV
D	90-13709	C

SCALE NONE SHEET 1 OF 2

REV	DESCRIPTION	ECO	DATE	APPROVED
1	PRE REL			
2	PRE REL			
3	PRE REL			
4	PRE REL			
A	REL TO PROD.			
B	INCORP	7446		
C	INCORPORATED	8864		



4. TO ISOLATE REGULATOR CONNECTION TO INVERTER, REMOVE JP1 & INSTALL IT TO JP2 POSITION TO KEEP REGULATOR VOLTAGE FROM RISING EXCESSIVELY.
3. DES. NOS. ON BOARD 1 (BOT) ARE ASSIGNED 100 ON, BOARD 2 (TOP) ARE ASSIGNED NOS. 1-99, EXCEPT FOR A, E, F, JP, P, S, AND T DES. WILL CON'T.
2. SHEET 1 AND PART OF SHT. 2 OF SCHEMATIC ARE BD1 (BOTTOM) SHEET 2 OF SCHEMATIC IS BD 2 (TOP)
1. UNLESS OTHERWISE SPECIFIED:
   RESISTORS ARE IN OHMS, ±5%, 1/4W
   CAPACITORS ARE IN MICROFARADS (UF)
   DIODES ARE IN4448
   VOLTAGE RATINGS ARE DC

NOTES:

BRD. NO. 2 (TOP)

REFERENCE DESIGNATORS LAST USED	NOT USED
A1	
C22	
CR27	
E4	
JP2	
L2	
P15	P1 - P11
Q8	
R30	
T2	T1
U2	

ITEM	QTY	PART NUMBER	DESCRIPTION	REF DESIGNATION

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES:
DECIMALS (.XX) ± FRACTIONS
DECIMALS (.XXX) ± ANGLES

NEXT ASSY	MOD NO.
45-13729	XL400-5411R

DO NOT SCALE DRAWING

APPROVALS: DATE
DRAWN
CHECKED
ENG
ENG MGR
MFG MGR
QA MGR
RELEASED

BOSCHERT ASSOCIATES SUNNYVALE, CALIFORNIA

SCHEMATIC
MODEL XL400-5411R

SIZE D | DWG NO. 90-13709 | REV C
SCALE NONE | SHEET 2 OF 2

REV	DESCRIPTION	ECO	DATE	APPROVED
1	PRE-REL.			
2	PRE REL			
3	PRE-REL		2-23-83	
4	PRE REL			
A	[illegible]		[illegible]	
B	INCORPORATED	7825	11/14/83	
C	INCORPORATED	8342A		

DWG NO 90-3076 REV

+ INPUT

J1 1

C1 470 50V
C11 470 50V
R2 620
R1 1.2K
Q1 MPS-A56
Q2 MPS A56
C2 .001 1KV
R3 0.1, 2W BWH
R4 0.1, 2W BWH
Q3 D44VH4
Q4 D45C11
R9 10, 1W BW20F 10%
R8 100
Q5 MPS-A56
R10 470
L1 80-11969
C10 0.1 100V
CR2 USD840 (43-11730-20)
C3 470 50V
R11 1K 2W BWH 10%
C14 470 50V
C15 270pf 1KV
C16 470 50V
C17 470 50V
L3 80-12209
L2 80-11762
J1 11
12 OUTPUT +
14 RETURN -
15
9 M-LOST
R56 4.7K
CR11
R19 8.2K
R55 20K
Q9 MPS-A56
C4 4.7-50V
C5 0.1 100V
R34 390K
R7 1.2K
C12 .0022-1KV
Q6 MPS-A06
Q8 MPS 5172
R27 5.6K
C8 .001 100V
R24 6.2K
R32 390K
R14 8.2K 2%
R39 47
U1 723
CW R13 2K
R12 43K 2%
C23 0.1 100V
R15 51K 2%
C18 .001 1KV
R16 2.4K 2%
R20 68K
C7 470pf 1KV
R17 100K
C6 .0022 1KV
Q15 MPS-A06
R18 2K
R57 10K
R38 4.3K
R21 10K
R22 10K
R23 1K
Q7 MPS-A06
CR5
CR3
CR4
R28 100K
R30 33K
R29 1K
C9 .001 100V
R31 270K 2%
R52 680
R51 3.3K
R50 5.6K
R49 5K
R48 18K
R53 39K
R54 4.7K
LM393N
U3(A1)
U3(B)
C21 1 50V
Q14 MPS2222A
C22 10 16V
VR1
TL430
7 ENABLE
CR9

CHG. SOURCE 3
TO BATT 5

C24 3.3 50V
R40 1.2K
R41 5K
U4 723
R42 1K
R43 8.2K
CR1 IN4004
Q12 TIP31
Q13 MPS2222 A
R47 10K
R46 6.8K
C20 .01 100V
R45 10K
R44 3.3

REFERENCE DESIGNATORS	
LAST USED	NOT USED
C24	C19,13
CR11	CR6,7,8,10
J1	
L3	
Q15	Q10,11
R57	R5,6,25,26,35-37, 33
U4	
VR1	

NOTES:
1. UNLESS OTHERWISE SPECIFIED:
CAPACITORS ARE IN MICROFARADS.
RESISTORS ARE IN OHMS, 1/4 WATT. C.F., 5%
DIODES ARE IN4448.
RATINGS ARE IN D.C.

ITEM	QTY	PART NUMBER	DESCRIPTION	REF DESIGNATION

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES:
DECIMALS (XX) — FRACTIONS —
DECIMALS (XXX) — ANGLES —

NEXT ASSY	MOD NO
	BB500

DO NOT SCALE DRAWING

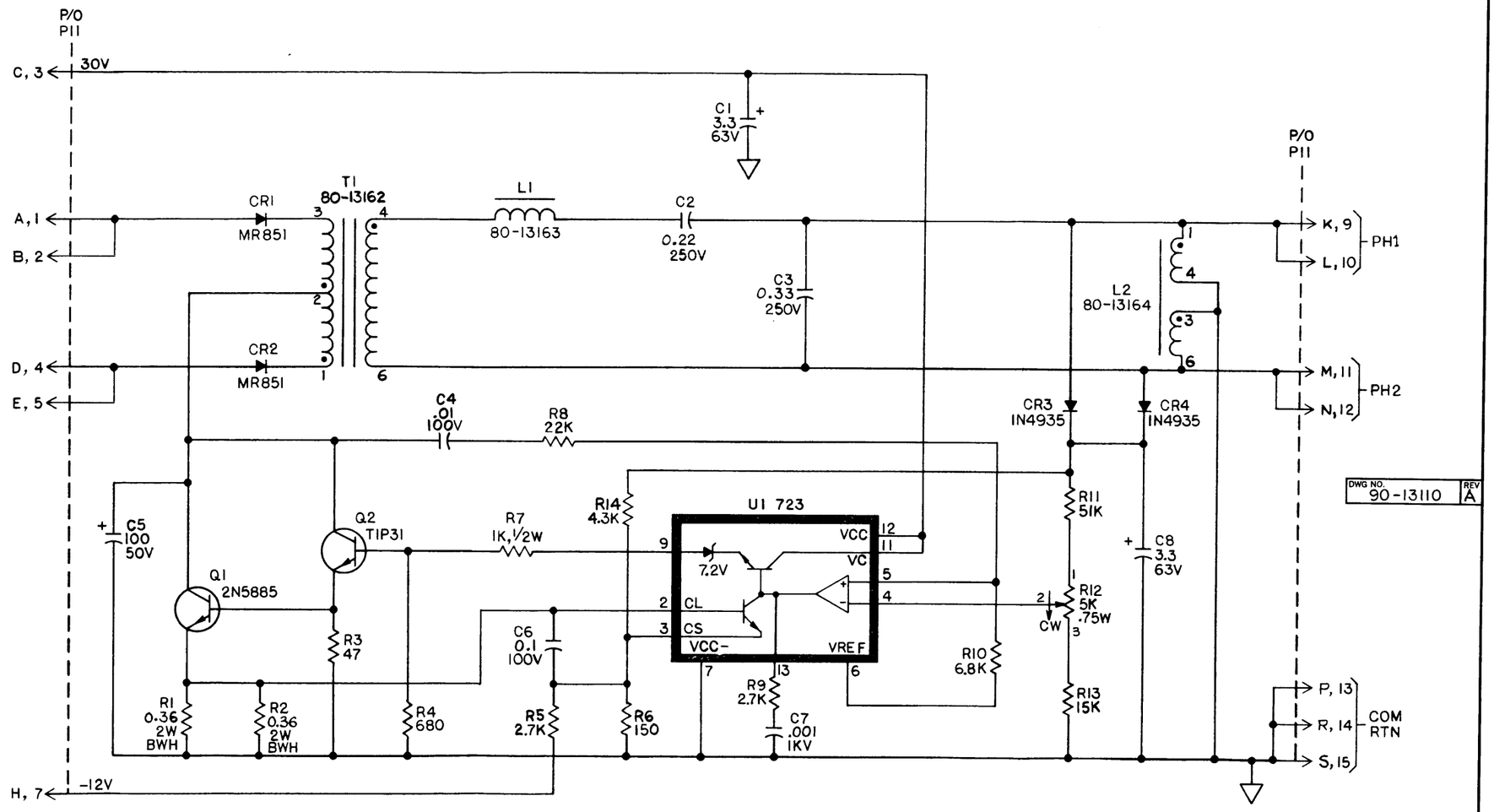
BOSCHERT ASSOCIATES SUNNYVALE CALIFORNIA

SCHEMATIC
+5.1V BATTERY BACK-UP SUPPLY
MODEL: BB500

SIZE	DWG NO	REV
D	90-13076	C
SCALE NONE	SHEET 1 OF 1	

90-13076-3

REV	DESCRIPTION	ECO	DATE	APPROVED
4	PRE REL FOR PROD		10/14/82	DD MC
5	PRE REL		12/1/82	MC LL JH
A	REL FOR PROD		1/20/83	MC DD



LAST USED	UNUSED
CR4	
Q2	
U1	
C8	
R14	
T1	
L2	

1. ALL VOLTAGES ARE "DC".
   ALL CAPACITORS ARE IN MICROFARADS
   ALL RESISTORS ARE IN OHMS, ± 5%, 1/4 W.

NOTES: UNLESS OTHERWISE SPECIFIED,

ITEM	QTY	PART NUMBER	DESCRIPTION	REF DESIGNATION

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES:
TOLERANCES:
DECIMALS (.XX) ± FRACTIONS ±
DECIMALS (.XXX) ± ANGLES ±

NEXT ASSY	MOD NO.
43-13728	XL400-5411

DO NOT SCALE DRAWING

BOSCHERT INCORPORATED
SUNNYVALE, CALIFORNIA

SCHEMATIC
SINE WAVE SUPPLY
SW100

SIZE	DWG NO.	REV
C	90-13110	A

SCALE N/A SHEET 1 OF 1

REV	DESCRIPTION	ECO	DATE	APPROVED
A	PRE REL			
A	REL FOR PROD			

ITEM	QTY	PART NUMBER	DESCRIPTION	REF DESIGNATION

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES:
DECIMALS (XX) ± FRACTIONS ±
DECIMALS (XXX) ± ANGLES ±

NEXT ASSY	MOD NO.
45 - 14418	

DO NOT SCALE DRAWING

Boschert SWITCHING POWER

SCHEMATIC
MODEL: XL301-5612 R

SIZE	DWG NO.	REV
D	90-14641	A
SCALE NA	SHEET 1 OF 2	

REV	DESCRIPTION	ECO	DATE	APPROVED

P/O T1 80-14729

L2 80-11944 / L7 80-12209

CR28 BYW29-150

R76 10 / C58 1000pf-160VDC/100VAC

C24 0.1 250V

C25 350 60V

R31 1K 2W-BWH 10%

SEE SHT1 FOR REF U3 P/O

U2 14-14984-01

R96 470 2W BWH

R108 1.3K 1W

84-14987-01 P4

9 +28V

6,7,8 +12V

R82 10

C67 1000pf 160VDC/100VAC

CR32 10-14482

L3 80-14034

C26 .022 63/100V

C27 570 40V

R122 120 2W BWH

R32 120 2W BWH

C68 1000pf 160VDC/100VAC

R83 10

R44 1.8K 1/2W

R42 27K

R116 10K

CR47 IN 5993B 5.1V

CR48 IN 5993B 5.1V

RJ1 ZERO HM

C69 0.1 63/100V

R98 9.1K

C70 0.1 63/100V

R99 2.55K 1/8W 1%

U7 LM393N

R100 10K

R101 10K

C60 1000pf-160 VDC/100VAC R78 10

CR30 BYV27-100

C64 1000pf-160 VDC/100VAC R79 10

CR31 BYV27-100

T3 80-14923

R102 680K

R103 2K

C71 0.1 63/100V

C59 1000pf -160VDC/ 100VAC R77 10

CR29 BYW29-150

CR21 IN5266B 68V

R26 330

Q9 TIP140

R27 180

C16 0.22 63V

R39 3.3K

C31 0.1-16V

R45 10K

VR2 TL431

R41 2K 0.5W

R40 6.8K

L4 80-14033

CR33 BYV28-150 (2PL)

CR34

C28 0.22 63V

C29 570 40V

R33 120 2W BWH

R123 120 2W BWH

R104 2K

C72 0.1 63/100V

R109 5.1K

5 -12V

JP2 RED

WHT

C65 1000pf-160/100VAC R80 10

CR45 BYV27-100

T4 80-14923

R72 1.8K 1/2W

R73 3.3K

C30 0.1-16V

CR22 IN5266B 68V

R38 330

Q15 TIP140

R28 180

C17 0.22 63V

VR3 TL431

R47 10K

R74 2K 0.5W

R37 27K

R75 6.8K

R105 680K

U7 LM393

CR51

CR52

R111 100K

CD4093 BE U5

10-23 SEC RET

CR36 SD51 45V

R34 10 C47 0.1 100V

CR46 BYV27-100

C66 1000pf-160VDC/ 100VAC R81 10

BLK E6 E7

R106 2.26K ±1% 1/8W

C74 0.1 63/100V

C73 0.1-63/100V

R107 2.55K 1/8W 1%

JP3

RED

CR37 SD-51-45V

C41 0.1 100V

R35 10

L5 80-15115

E8 E9 E10 E11 E13 E12

C32 570 40V

C33 570 40V

C34 570 40V

C35 1600 15V

R120 10 5W

R43 10 5W

C39 0.1 250V

3 25KHZ

4 25 KHZ

24-35 +5.1V

R68 10K

CD4093 BE U5

Q12 VN10KM

1 PON

R70

SEE SHT1 FOR REF

P/O U1 89-12977

L6 80-10799

R49 1.2

R52 330

C40 0.22 63V

R53 470

R54 2.4K

R63 10K

R65 5K 0.5W

R14 2.4K

VR1 TL431

C48 10 100V

R110 820K

CR49

CR50

C75 0.1 63/100V

C76 0.1 63/100V

U8 CD4538 BE

R112 2K

R113 332K 1/8W 1%

C77 .022-63/100V

R115 909K 1/8W,1%

C78 .022 63V

R121 2M

R114 5.1K

R69 10K

Q13 VN10KM

CR53

C83 1 63V

2 PFW

REFERENCE DESIGNATORS	
LAST USED	NOT USED
C87	C20, 36, 50, 57, 42, 49, 61, 79, 82, 86
CR 53	CR5, 14, 19, 20, 23, 27, 35, 44
E13	
F2	
JP4	
Q21	Q11, Q16
L9	
R124	R 17, 24, 29, 36, 48, 71, 97, 118, 119
RT2	
S1	
U9	
VR3	
RJ1	
P4	
T4	

4 C55 & C56 NOT USED WHEN 1700 uf MEPCO CAPS, C43-C46 ARE USED.

3 FOR UTILIZATION OF JP1 SEE TEST PROCEDURE 91-51461.

2 U1 & R70 ARE A MATCHED SET.

VOLTAGES ARE D.C.
DIODES ARE IN4448
CAPACITORS ARE IN MICROFARADS
RESISTORS ARE IN OHMS, ± 5%, 1/4 W.

NOTES: 1. UNLESS OTHERWISE SPECIFIED;

ITEM	QTY	PART NUMBER	DESCRIPTION	REF DESIGNATION

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES:
TOLERANCES:
DECIMALS (.XX) ± FRACTIONS ±
DECIMALS (.XXX) ± ANGLES ±

NEXT ASSY	MOD NO.

DO NOT SCALE DRAWING

APPROVALS: DATE
DRAWN
CHECKED
ENG.
ENG MGR
MFG MGR
QA MGR
RELEASED

Boschert SWITCHING POWER

SCHEMATIC
XL 301-5612 R

SIZE D | DWG NO 90-14641 | REV A

SCALE N/A | SHEET 2 OF 2

A600+ Baseset Microcode	APPENDIX D

A representative listing of the baseset microcode used in the A600+ computer is presented in the following pages. This listing may not accurately reflect the as-installed configuration of microcode actually contained in ROM. The listing is solely intended to serve as an aid to those users attempting to develop microprograms.

```
LINE    ADDR    STATEMENT

   1            TITLE       A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]
   2            ;                                                    <840731.1350>
   3            ;   REVISION HISTORY
   4            ; MICROCODE REV 0 - AS ISSUED
   5            ; MICROCODE REV 1 - FIXED AB REFERENCE IN CROSS MAP INSTRUCTIONS
   6            ;                   (AFFECTS VALUES IN MAP FILE)
   7            ; (NO REV 2)
   8            ; MICROCODE REV 3 - FIXED JPY TO NOT ADD BASE REGISTER    JULY 28, 1983
   9            ;
  10            NOLIST  L
  11            LIST    B,E
  12            ;
  13            ;*****************************************************************
  14            ;**                                                             **
  15            ;**        A600+ Baseset Microcode                              **
  16            ;**                                                             **
  17            ;**          "Thunder is good, Thunder is impressive,           **
  18            ;**           but it was Lightning that did the work"           **
  19            ;**                                                             **
  20            ;**  TDI:                                                       **
  21            ;**        - Temporary Disable Interrupt is set to inhibit      **
  22            ;**          interrupts at the conclusion of JMP.I, JSB.I, and  **
  23            ;**          I/O instructions so that the next instruction      **
  24            ;**          will be executed.                                  **
  25            ;**        - The ENCN SETTDI microorder must be given at least  **
  26            ;**          one cycle before IFETCH.  TDI is automatically     **
  27            ;**          cleared one cycle after JTAB INIT (Line 0) and by  **
  28            ;**          the ENCN SETMPI microorder.                        **
  29            ;**                                                             **
  30            ;**  CARRY:                                                     **
  31            ;**        - The Cn input to the ALU is XOR'D with I3*/MRG .     **
  32            ;**          Thus, the sense of carry is inverted for SUBR, OR, **
  33            ;**          NOTRS and EXNOR, for non-MRG instructions.  This   **
  34            ;**          means that SUBR should not be used for interrupt   **
  35            ;**          routines, as the IR contents are indeterminate.    **
  36            ;**                                                             **
  37            ;**  TAB Special:                                               **
  38            ;**        - In A reg field, modifies source operand of DZ to   **
  39            ;**          ZA (DA to AB, AQ to DQ), and selects A operand of  **
  40            ;**          macro A or B register depending on Y0BUF if macro  **
  41            ;**          A or B register was addressed on last DREAD or     **
  42            ;**          IFETCH (MRGREAD or MRGFETCH).  If the last value   **
  43            ;**          loaded into the MAR was not 0 or 1, the source     **
  44            ;**          operand field is not modified, and the A operand   **
  45            ;**          is either the macro A or B register depending on   **
  46            ;**          IR11.                                              **
  47            ;**                                                             **
  48            ;**        - In B reg field, modifies destination operand of    **
  49            ;**          NOP to RAMF and selects a B operand of macro A or  **
  50            ;**          B reg depending on Y0BUF, when the MAR contains a  **
  51            ;**          0 or 1.  Otherwise, the destination operand is     **
  52            ;**          unmodified.                                        **
  53            ;**                              *** NOTE ***                   **
  54            ;**        - TAB may not be used as a DESTINATION (B field)     **
```

```
LINE    ADDR    STATEMENT

  55            ;**         special in the same line as LDMAR                **
  56            ;**                                                          **
  57            ;**  BR Special:                                             **
  58            ;**       - Chooses Q as the A operand                       **
  59            ;**       - If last memory read was from A or B, changes     **
  60            ;**         source operand from DZ to ZB                     **
  61            ;**       - If the new address is base relative (in range 2 to **
  62            ;**         1777 octal inclusive), and CSMODE is ON, freeze  **
  63            ;**         for one cycle and change DZ to DA (ZB to AB),    **
  64            ;**         which has the result of adding the base register.  **
  65            ;**                                                          **
  66            ;**  TABQ special:                                           **
  67            ;**       - Selects macro A or B register depending on the   **
  68            ;**         signal SRCY0BUF (if the last memory reference was **
  69            ;**         to A or B).                                      **
  70            ;**                                                          **
  71            ;**         The BR special and the TABQ special work together  **
  72            ;**         to provide proper addresses (A/B/Base relative)  **
  73            ;**         during indirect resolution.                      **
  74            ;**                                                          **
  75            ;**  PORM special:                                           **
  76            ;**       - Chooses between PC and MP based on the signal    **
  77            ;**         PCA1 from the IRDECODE FPLA.  At LINE1, PCA1 is  **
  78            ;**         asserted (choose MP) for the instructions        **
  79            ;**         JSB,D JMP,D ISZ,D and ST*,D.                     **
  80            ;**                                                          **
  81            ;**  PORQ special:                                           **
  82            ;**       - Chooses between PC and Q (base register) based on **
  83            ;**         the signal PCA0 from the IRDECODE FPLA.  At LINE0, **
  84            ;**         PCA0 is asserted (choose Q) for all MRG          **
  85            ;**         instructions.                                    **
  86            ;**       - If PC is chosen, then change source operand of DZ **
  87            ;**         to ZA.  If Q is chosen, then change DZ to DA if  **
  88            ;**         CSMODE is ON and IR bit 10 is 0 and the reference **
  89            ;**         is not to A or B (base relative MRG).            **
  90            ;**                                                          **
  91            ;**  MPY special:                                            **
  92            ;**       - Selects a source operand of AB or ZB depending on **
  93            ;**         the signal Q0BUF (Q0BUF+ HIGH selects AB).  MPY  **
  94            ;**         puts the result in register 0 (A register), and  **
  95            ;**         MPY8 puts the result in register 8 (S0).         **
  96            ;**                                                          **
  97            ;**  DIV special:                                            **
  98            ;**       - Changes an ALU function of SUBR to ADD depending **
  99            ;**         on the condition code output from the last       **
 100            ;**         microcycle (normally DIVCOND = dividend sign XOR **
 101            ;**         divisor sign).                                   **
 102            ;**                                                          **
 103            ;**  SEQFRZ special:                                         **
 104            ;**       - Freezes the ALU and the sequencer (but not the   **
 105            ;**         AM2904 status unit) for one cycle.  This has the **
 106            ;**         effect that you can set and test a microcode     **
 107            ;**         condition in the same line.  The SEQFRZ special  **
 108            ;**         freezes UCLK- LOW.                               **
```

```
A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]     1:35 PM  TUE., 13  SEPT, 1983 PAGE   3
LINE    ADDR    STATEMENT

 109            ;**                                                              **
 110            ;**************************************************************************

A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]     1:35 PM  TUE., 13  SEPT, 1983 PAGE   4
LINE    ADDR    STATEMENT

 112            ;**************************************************************************
 113            ;**                                                              *
 114            ;**       EQUATES                                                *
 115            ;**                                                              *
 116            ;**************************************************************************
 117            ;
 118            IMAPLOC:  EQU   H#0041     ; BOOT MEM ADDRESS OF IMAP REG (101Q)
 119            VMALOC:   EQU   H#0004     ; RTE VMA FAULT TRAP CELL
 120            VMAPTE:   EQU   H#0002     ; VMA PAGE TABLE POINTER LOCATION
 121            IQLOC:    EQU   H#0042     ; BOOT MEMORY ADDRESS OF INTD_Q (102Q)
 122            SEGTRAP:  EQU   H#000B     ; RTE SEGMENT FAULT TRAP CELL (13 OCT)
 123            VMATRAP:  EQU   H#000A     ; RTE VMA TRAP CELL FOR CDS ON (12 OCT)
 124            ;
 125            CPUID:    EQU   H#0005     ; THUNDER PROCESSOR ID NUMBER
 126            MICREVID: EQU   H#0003     ; MICRO CODE REVISION NUMBER
 127            ;
 128            EXIT0:    EQU   H#8B0F     ; .EXIT0 OPCODE
 129            EXIT1:    EQU   H#8B0D     ; .EXIT1 OPCODE
 130            EXIT2:    EQU   H#8B0E     ; .EXIT2 OPCODE
 131            ;
```

```
LINE    ADDR    STATEMENT

133             ;*********************************************************************
134             ;*                                                                   *
135             ;*         Basic Instruction Decode                                  *
136             ;*         ------------------------                                  *
137             ;*                                                                   *
138             ;* - At the FETCH line, PC points to Next instruction.               *
139             ;*                                                                   *
140             ;* - After instruction mapping, MAR points to the next               *
141             ;*   instruction, or to the second word of multi-word                *
142             ;*   instructions, and PC points to the current instruction + 2      *
143             ;*                                                                   *
144             ;* - First microinstruction:                                         *
145             ;*         - Wait for instruction fetch to finish                    *
146             ;*         - Load MAR and scratch reg MP with MRG address            *
147             ;*           (for MRG instructions) or with PC                       *
148             ;*         - Start read of MRG address or second word (multi-        *
149             ;*           word instructions)                                      *
150             ;*         - Load 2910 register with uaddr of instruction routine    *
151             ;*                                                                   *
152             ;* - Second microinstruction:                                        *
153             ;*         - Increments PC                                           *
154             ;*         - IF ISZ, ST*, JMP, or JSB THEN load MAR with MRG         *
155             ;*           address ELSE load MAR with next instruction address     *
156             ;*           or current instruction second word address.             *
157             ;*         - IF interrupt pending THEN jump to interrupt handler     *
158             ;*           ELSE jump to instruction microcode routine.             *
159             ;*                                                                   *
160             ;*********************************************************************
161             ;*****
162             ;*****    - Wait for instruction to become valid in IR
163             ;*****    - Load MAR and register MP with MRG actual address if
164             ;*****        MRG, or with PC (=next_inst) if not MRG
165             ;*****    - Put mapped address in 2910 register
166             ;*****    - Do a read if MRG read instruction or multi-word inst.
167             ;*****
168             FETCH:    LDCT              & AM2901 PORQ,MP,RAMF,ADD,DZ
169             /                           & CARRYL          ; no carry for add
170             /                           & JTAB INIT       ; Initial instruction decod
171             /                           & LDMAR           ; Load MAR
172   00000     /                           & MWAIT           ; Wait for FETCH
173             ;
174             ;*****    - Load MAR with PC or MP, depending on whether you
175             ;*****        are executing a PCA1 type instruction or not
176             ;*****    - Increment PC
177             ;*****    - Jumps to the appropriate microcode routine for the
178             ;*****        instruction being mapped, or INTERRPT if there is a
179             ;*****        qualified interrupt pending.
180             ;*****
181                       JRP   INTERRPT    & AM2901 PORM,PC,RAMA,ADD,ZB
182             /                           & CARRYH          ; Inc PC
183             /                           & CONDEXT INTRPT  ; JP if int pend, else JR
184             /                           & LODMSR          ; Load YCBUF for ASCONT
185   00001     /                           & LDMAR           ; Load MAR with PC or MP
```

```
LINE    ADDR    STATEMENT

 187            ;*************************************************************************
 188            ;*                                                                       *
 189            ;*        Memory Reference Group                                         *
 190            ;*        ----------------------                                         *
 191            ;*                                                                       *
 192            ;*        - Resolved address is in QREG                                  *
 193            ;*        - MAR contains PC-1 (=next_inst) for MRG read                  *
 194            ;*          instructins                                                  *
 195            ;*                                                                       *
 196            ;*        MEM is the resolved MRG address.                               *
 197            ;*                                                                       *
 198            ;*************************************************************************
 199            ;*****
 200            ;*****   - Add (MEM) to A/B.  MEM is A/B addressable
 201            ;*****
 202            AD.I:     CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
 203            /                          & CARRYL         ;
 204            /                          & LODUSR         ;
 205            /                          & LDMAR          ;
 206   00002    /                          & DREAD          ;
 207            AD.:      JZ               & AM2901 TAB,CAB,RAMF,ADD,DA
 208            /                          & CARRYL         ;
 209            /                          & LODMSR ENVE    ;
 210            /                          & ENBLC & ENBLO  ;
 211   00003    /                          & IFETCH         ;
 212            ;*****
 213            ;*****   - AND (MEM) and A/B.  MEM is A/B addressable
 214            ;*****
 215            ANDI:     CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
 216            /                          & CARRYL         ;
 217            /                          & LODUSR         ;
 218            /                          & LDMAR          ;
 219   00004    /                          & DREAD          ;
 220            AND.:     JZ               & AM2901 TAB,A,RAMF,AND,DA
 221   00005    /                          & IFETCH         ;
 222            ;*****
 223            ;*****   - Compare A/B with (MEM).  MEM is A/B addressable
 224            ;*****   - If the two operands are identical, execute the
 225            ;*****       next instruction, else skip (do if true).
 226            ;*****
 227            CP.I:     CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
 228            /                          & CARRYL         ;
 229            /                          & LODUSR         ;
 230            /                          & LDMAR          ;
 231   00006    /                          & DREAD          ;
 232            CP.:      JP  SKIFNZ       & AM2901 TAB,CAB,NOP,EXOR,DA
 233   00007    /                          & LODUSR         ;
 234            ;*****
 235            ;*****   - OR (MEM) with A/B.  MEM is A/B addressable
 236            ;*****
 237            IORI:     CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
 238            /                          & CARRYL         ;
 239            /                          & LODUSR         ;
```

```
LINE   ADDR   STATEMENT

 240          /                          & LDMAR          ;
 241   00008  /                          & DREAD          ;
 242          IOR:     JZ                & AM2901 TAB,A,RAMF,OR,DA
 243   00009  /                          & IFETCH         ;
 244          ;*****
 245          ;*****   - Increment (MEM) and skip next instruction if the
 246          ;*****       result is zero.
 247          ;*****   - MEM is A/B addressable.
 248          ;*****
 249          ISZI:    CCALL READRES     & AM2901 BR,TABQ,QREG,ADD,DZ
 250          /                          & CARRYL         ;
 251          /                          & CONDEXT SIGN   ;
 252          /                          & LDMAR          ;
 253   0000A  /                          & DREAD          ;
 254          ISZ:     CONT              & AM2901 TAB,TAB,NOP,ADD,DZ
 255          /                          & CARRYH         ;
 256          /                          & LODMSR         ;
 257          /                          & LDMDOR         ;
 258   0000B  /                          & DWRITE         ; LOAD MSR FOR NEXT LINE
 259                   JP    SKIFZ       & AM2901 ,PC,NOP,SUBR,ZB
 260          /                          & CARRYL         ;
 261          /                          & LODMSR SWAPUM  ;
 262   0000C  /                          & LDMAR          ; USR = STATUS FROM PREV. L
 263          ;*****
 264          ;*****   - Jump indirect.
 265          ;*****   - Sets TDI so that the jump target instruction is
 266          ;*****       always executed after the jump.
 267          ;*****
 268          JMPI:    CCALL  INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
 269          /                          & CARRYL         ;
 270          /                          & CONDEXT SIGN   ;
 271          /                          & LDMAR          ;
 272   0000D  /                          & CMSGN & DREAD  ; RESOLVE INDIRECTS
 273          JPTDI:   CONT              & AM2901 ,,NOP,ADD,AQ
 274   0000E  /                          & ENCN SETTDI    ; TEMP INTERRUPT DISABLE
 275          INCFTCH: JZ                & AM2901 ,PC,RAMF,ADD,ZQ
 276          /                          & CARRYH         ;
 277   0000F  /                          & IFETCH         ; INC PC AND FETCH NEXT INS
 278          ;*****
 279          ;*****   - Jump direct
 280          ;*****
 281          JMP:     JZ                & AM2901 MP,PC,RAMF,ADD,ZA
 282   00010  /                          & CARRYH         ; INC PC
 283          ;*****
 284          ;*****   - JSB indirect
 285          ;*****   - Sets TDI so that the target of the JSB is always
 286          ;*****       executed.
 287          ;*****
 288          JSBI:    CCALL  INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
 289          /                          & CARRYL         ;
 290          /                          & CONDEXT SIGN   ;
 291          /                          & LDMAR          ;
 292   00011  /                          & CMSGN & DREAD  ;
 293                   CONT              & AM2901 PC,TAB,NOP,SUBR,ZA
```

```
LINE   ADDR    STATEMENT

 294           /                          & CARRYL          ;
 295           /                          & LDMDOR          ;
 296   00012   /                          & DWRITE          ; SUBTRACT 1 FROM PC
 297           JSBIVMA:  JP  SKIP         & AM2901 ,PC,RAMF,ADD,ZQ
 298           /                          & CARRYH          ;
 299   00013   /                          & ENCN SETTDI     ; SET PC TO JSB TARGET + 1
 300           ;*****
 301           ;*****    - JSB direct
 302           ;*****    - Since PC points to Next instruction + 1, PC-1 must
 303           ;*****        be stored at JSB target.
 304           ;*****
 305           JSB:      CONT             & AM2901 PC,TAB,NOP,SUBR,ZA
 306           /                          & CARRYL          ;
 307           /                          & LDMDOR          ;
 308   00014   /                          & DWRITE          ; SUBTRACT 1
 309                     JP    SKIP       & AM2901 MP,PC,RAMF,ADD,ZA
 310   00015   /                          & CARRYH          ;
 311           ;*****
 312           ;*****    - Load A/B with (MEM).  MEM is A/B addressable
 313           ;*****
 314           LD.I:     CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
 315           /                          & CARRYL          ;
 316           /                          & LODUSR          ;
 317           /                          & LDMAR           ;
 318   00016   /                          & DREAD           ;
 319           LD.:      JZ               & AM2901 TAB,CAB,RAMF,PASS,DZ
 320   00017   /                          & IFETCH          ;
 321           ;*****
 322           ;*****    - Store A/B to MEM.  MEM is A/B addressable
 323           ;*****
 324           ST.I:     CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
 325           /                          & CARRYL          ;
 326           /                          & CONDEXT SIGN    ;
 327           /                          & LDMAR           ;
 328   00018   /                          & CMSGN & DREAD   ;
 329           ST.:      CONT             & AM2901 CAB,TAB,NOP,PASS,ZA
 330           /                          & LDMDOR          ;
 331   00019   /                          & DWRITE          ;
 332                     JZ               & AM2901 ,PC,NOP,SUBR,ZB
 333           /                          & CARRYL          ; PC - 1
 334           /                          & LDMAR           ;
 335   0001A   /                          & IFETCH          ;
 336           ;*****
 337           ;*****    - Exclusive-OR (MEM) with A/B.  MEM is A/B addressable
 338           ;*****
 339           XORI:     CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
 340           /                          & CARRYL          ;
 341           /                          & LODUSR          ;
 342           /                          & LDMAR           ;
 343   0001B   /                          & DREAD           ;
 344           XOR:      JZ               & AM2901 TAB,A,RAMF,EXOR,DA
 345   0001C   /                          & IFETCH          ;
```

```
LINE    ADDR    STATEMENT

347             ;*********************************************************************
348             ;*                                                                   *
349             ;*        Shift Rotate Group                                         *
350             ;*        ------------------                                         *
351             ;*                                                                   *
352             ;*  The initial instruction decode jumps to the SRG0 entry           *
353             ;*  points.                                                          *
354             ;*                                                                   *
355             ;*  Specials:                                                        *
356             ;*                                                                   *
357             ;*  SRG1 - Maps bits IR5 and IR3 to control store address            *
358             ;*         to decode CLE, SL*, and NOP                               *
359             ;*       - SRG1 performs the IFETCH                                  *
360             ;*      !! Depends on CONDEXT IRSKPBUF and LODMSR RESET              *
361             ;*         having bits PL12-PL15 the same                            *
362             ;*                                                                   *
363             ;*  SRG2 - Maps bottom 6 bits of IR to control store address         *
364             ;*       - Decodes second Shift/Rotate operation:                    *
365             ;*         *LS,*RS,R*L,R*R,*LR,ER*,EL*,*LF,EL*D,ER*D                 *
366             ;*                                                                   *
367             ;*********************************************************************
368             ;*****
369             ;*****    - Rotate left 4
370             ;*****    - RL4 special puts YBUS rotated left 4 on the DBUS
371             ;*****
372             SRG0.LF:  JMAP  SRG1        & AM2901 CAB,CAB,RAMA,PASS,DZ
373   0001D     /                           & SPRD RL4        ;
374             ;*****
375             ;*****    - Left shift one, clear sign
376             ;*****    - Algorithm is :  Mask off bits 14 and 15
377             ;*****                      Shift left normal
378             ;*****
379             SRG0.LR:  CONT              & AM2901 ,,QREG,EXNOR,DZ
380   0001E     /                           & IMM H#C000      ; QREG = H#3FFF
381                       JMAP  SRG1        & AM2901 CAB,CAB,SRAML,AND,AQ
382   0001F     /                           & SHIFT B#0010    ;
383             ;*****
384             ;*****    - Arithmetic shift left
385             ;*****    - Algorithm is :  Set sign and left shift
386             ;*****                      Left shift again
387             ;*****                      Right shift with sign (restore sign)
388             ;*****
389             SRG0.LS:  CONT              & AM2901 ,CAB,SRAML,PASS,ZB
390             /                           & LODMSR          ;
391   00020     /                           & SHIFT B#0010    ;
392                       CONT              & AM2901 ,CAB,SRAML,PASS,ZB
393   00021     /                           & SHIFT B#0010    ;
394                       JMAP  SRG1        & AM2901 ,CAB,SRAMR,PASS,ZB
395   00022     /                           & SHIFT B#0101    ;
396             ;*****
397             ;*****    - Arithmetic right shift
398             ;*****    - Algorithm is :  Set sign
399             ;*****                      Right shift with sign
```

```
LINE    ADDR    STATEMENT

 400            ;*****
 401            SRG0.RS:  CONT            & AM2901 ,CAB,NOP,PASS,ZB
 402   00023    /                         & LODMSR         ;
 403                      JMAP  SRG1       & AM2901 ,CAB,SRAMR,PASS,ZB
 404   00024    /                         & SHIFT B#0101   ;
 405            ;*****
 406            ;*****    - Rotate left with E
 407            ;*****
 408            SRG0EL.:  JMAP  SRG1       & AM2901 ,CAB,SRAML,PASS,ZB
 409   00025    /                         & SHIFT ROTATEC  ;
 410            ;*****
 411            ;*****    - Copy sign into E
 412            ;*****    - Same as rotate left with E, except scratch destination
 413            ;*****
 414            SRG0EL.D: JMAP  SRG1       & AM2901 CAB,S0,SRAML,PASS,ZA
 415   00026    /                         & SHIFT ROTATEC  ;
 416            ;*****
 417            ;*****    - Rotate right with E
 418            ;*****
 419            SRG0ER.:  JMAP  SRG1       & AM2901 ,CAB,SRAMR,PASS,ZB
 420   00027    /                         & SHIFT ROTATEC  ;
 421            ;*****
 422            ;*****    - Copy LSB into E
 423            ;*****    - Same as rotate right with E, except scratch dest.
 424            ;*****
 425            SRG0ER.D: JMAP  SRG1       & AM2901 CAB,S0,SRAMR,PASS,ZA
 426   00028    /                         & SHIFT ROTATEC  ;
 427            ;*****
 428            ;*****    - NOP in first SRG position
 429            ;*****
 430            SRG0NOP:  JMAP  SRG1       & AM2901 CAB,,NOP,PASS,ZA
 431   00029    /                         & SPNOP          ; PUT A/B ON YBUS FOR SRG1
 432            ;*****
 433            ;*****    - Rotate left
 434            ;*****
 435            SRG0R.L:  JMAP  SRG1       & AM2901 ,CAB,SRAML,PASS,ZB
 436   0002A    /                         & SHIFT ROTATE   ;
 437            ;*****
 438            ;*****    - Rotate right
 439            ;*****
 440            SRG0R.R:  JMAP  SRG1       & AM2901 ,CAB,SRAMR,PASS,ZB
 441   0002B    /                         & SHIFT ROTATE   ;
 442            ;*****
 443            ;*****    - Perform SRG CLE/SL* operation
 444            ;*****
 445            SR1CLE:   JMAP  SRG2       & AM2901 CAB,,NOP,PASS,ZA
 446            /                         & LODMSR RESET   ; PUT A/B ON YBUS (DIAG)
 447   0002C    /                         & ENBLC          ;
 448            SR1SL.:   CONT            & AM2901 ,CAB,NOP,PASS,ZB
 449   0002D    /                         & SPNOP ;        ; PUT A/B ON YBUS FOR IRSKP
 450            SL.CONT:  JMAP  SRG2       & AM2901 PC,PC,RAMA,ADD,ZB
 451            /                         & CARRYEXT       ;
 452            /                         & CONDEXT IRSKBF ;
 453   0002E    /                         & CLD & LDMAR    ;
```

```
LINE   ADDR   STATEMENT

 454          SR1CLESL: JP    SL.CONT    & AM2901 ,CAB,NOP,PASS,ZB
 455          /                          & LODMSR RESET   ; PUT A/B ON YBUS FOR IRSKP
 456  0002F   /                          & ENBLC          ;
 457          ;*****
 458          ;*****    - Perform second SRG operation
 459          ;*****
 460          SRG2.LF:  JZ               & AM2901 CAB,CAB,RAMA,PASS,DZ
 461          /                          & SPRD RL4       ;
 462  00030   /                          & IFETCH         ;
 463          SRG2.LR:  CONT             & AM2901 ,,QREG,EXNOR,DZ
 464          /                          & IMM H#C000     ;
 465  00031   /                          & IFETCH         ;
 466                    JZ               & AM2901 CAB,CAB,SRAML,AND,AQ
 467  00032   /                          & SHIFT B#0010   ;
 468          SRG2.LS:  CONT             & AM2901 ,CAB,SRAML,PASS,ZB
 469          /                          & LODMSR         ;
 470  00033   /                          & SHIFT B#0010   ;
 471                    CONT             & AM2901 ,CAB,SRAML,PASS,ZB
 472          /                          & SHIFT B#0010   ;
 473  00034   /                          & IFETCH         ;
 474                    JZ               & AM2901 ,CAB,SRAMR,PASS,ZB
 475  00035   /                          & SHIFT B#0101   ;
 476          SRG2.RS:  CONT             & AM2901 ,CAB,NOP,PASS,ZB
 477          /                          & LODMSR         ;
 478  00036   /                          & IFETCH         ;
 479                    JZ               & AM2901 ,CAB,SRAMR,PASS,ZB
 480  00037   /                          & SHIFT B#0101   ;
 481          SRG2EL.:  JZ               & AM2901 ,CAB,SRAML,PASS,ZB
 482          /                          & SHIFT ROTATEC  ;
 483  00038   /                          & IFETCH         ;
 484          SRG2EL.D: JZ               & AM2901 CAB,S0,SRAML,PASS,ZA
 485          /                          & SHIFT ROTATEC  ;
 486  00039   /                          & IFETCH         ;
 487          SRG2ER.:  JZ               & AM2901 ,CAB,SRAMR,PASS,ZB
 488          /                          & SHIFT ROTATEC  ;
 489  0003A   /                          & IFETCH         ;
 490          SRG2ER.D: JZ               & AM2901 CAB,S0,SRAMR,PASS,ZA
 491          /                          & SHIFT ROTATEC  ;
 492  0003B   /                          & IFETCH         ;
 493          SRG2NOP:  JZ               & AM2901 ,,NOP,ADD,AQ
 494  0003C   /                          & IFETCH         ;
 495          SRG2R.L:  JZ               & AM2901 ,CAB,SRAML,PASS,ZB
 496          /                          & SHIFT ROTATE   ;
 497  0003D   /                          & IFETCH         ;
 498          SRG2R.R:  JZ               & AM2901 ,CAB,SRAMR,PASS,ZB
 499          /                          & SHIFT ROTATE   ;
 500  0003E   /                          & IFETCH         ;
```

```
LINE    ADDR    STATEMENT

 502            ;*********************************************************************
 503            ;*                                                                   *
 504            ;*        Alter-Skip Group                                           *
 505            ;*        ----------------                                           *
 506            ;*                                                                   *
 507            ;*   - ASG is mapped 4 ways based on CL*/CM*/CC*/NOP                 *
 508            ;*                                                                   *
 509            ;*   - Accumulator ops: CL* - Clears A/B to 0000H                    *
 510            ;*                      CC* - Sets A/B to 0FFFFH                     *
 511            ;*                      CM* - Inverts A/B (One's Complement)         *
 512            ;*                      NOP - No A/B register operation              *
 513            ;*                                                                   *
 514            ;*   - E-reg ops:       CLE - Clears E reg                           *
 515            ;*                      CCE - Sets E reg                             *
 516            ;*                      CME - Inverts E reg                          *
 517            ;*                      NOP - Hold E                                 *
 518            ;*                                                                   *
 519            ;*   - The E reg operations are done by the MSR special FPLA         *
 520            ;*       in the course of doing the A/B reg ops.                     *
 521            ;*                                                                   *
 522            ;*   - An accumulator NOP combined with an E reg NOP                 *
 523            ;*     maps directly to the ASCONT line.  All other                  *
 524            ;*     combinations map to the appropriate A/B/E                     *
 525            ;*     operation line.                                               *
 526            ;*                                                                   *
 527            ;*   - Algorithms:                                                   *
 528            ;*                                                                   *
 529            ;*        CL* - And register with zero with carry-in                 *
 530            ;*              LOW.   Cn+4 will be 0.                               *
 531            ;*        CC* - Subtract register from itself with carry-in          *
 532            ;*              LOW.   Cn+4 will be 0.                               *
 533            ;*        CM* - Subtract register from zero with carry-in            *
 534            ;*              LOW.   Cn+4 will be 0.                               *
 535            ;*                                                                   *
 536            ;*  !! The ASCONT line depends on the codes for LODMSR ENVE          *
 537            ;*     and CONDEXT IR2 being identical in bits PL12-PL15             *
 538            ;*                                                                   *
 539            ;*   - Note that PC already points to the target of the skip         *
 540            ;*                                                                   *
 541            ;*********************************************************************
```

```
LINE    ADDR    STATEMENT

 543            ;*****
 544            ;*****    - Clear and complement A/B (set).
 545            ;*****
 546            CC.:      JP    ASCONT      & AM2901 CAB,CAB,RAMF,SUBS,AB
 547            /                           & CARRYL          ;
 548   0003F    /                           & LODMSR ASGSP    ; LOAD YCBUF FOR ASCONT
 549            ;*****
 550            ;*****    - Clear A/B.
 551            ;*****
 552            CL.:      JP    ASCONT      & AM2901 ,CAB,RAMF,AND,ZB
 553   00040    /                           & LODMSR ASGSP    ; LOAD YCBUF FOR ASCONT
 554            ;*****
 555            ;*****    - Complement A/B.
 556            ;*****
 557            CM.:      JP    ASCONT      & AM2901 ,CAB,RAMF,SUBS,ZB
 558            /                           & CARRYL          ;
 559   00041    /                           & LODMSR ASGSP    ; LOAD YCBUF FOR ASCONT
 560            ;*****
 561            ;*****    - No A/B op, but set E appropriately.
 562            ;*****
 563            ASGEOP:   JP    ASCONT      & AM2901 ,CAB,NOP,ADD,ZB
 564            /                           & CARRYL          ;
 565   00042    /                           & LODMSR ASGSP    ; LOAD YCBUF FOR ASCONT
 566            ;*****
 567            ;*****    - Do IN* (Add bit 2 of IR) and set E & O appropriately
 568            ;*****    - NOPNOP maps here
 569            ;*****
 570            ASCONT:   CONT                & AM2901 CAB,CAB,RAMA,ADD,ZB
 571            /                             & CARRYEXT        ; IN* FUNCTION
 572            /                             & CONDEXT         ; CONDEXT IS IR2
 573            /                             & LODMSR ENVE     ; !! ENVE IS SAME AS CONDEX
 574   00043    /                             & ENBLC & ENBLO   ; AFFECTS OVERFLOW AND EXTE
 575            ;*****
 576            ;*****    - Do skip function
 577            ;*****
 578                      JZ                  & AM2901 PC,PC,RAMA,ADD,ZB
 579            /                             & CARRYEXT        ;
 580            /                             & CONDEXT IRSKBF  ;
 581            /                             & CLD & LDMAR     ;
 582   00044    /                             & IFETCH          ; SKIP FUNCTION
```

```
LINE    ADDR    STATEMENT

 584            ;*********************************************************************
 585            ;*                                                                   *
 586            ;*        Input - Output Group                                       *
 587            ;*        --------------------                                       *
 588            ;*                                                                   *
 589            ;*   - In the FETCH line, I/O instructions are mapped into           *
 590            ;*     three groups:  Halt, select code >= 200, and < 200.           *
 591            ;*                                                                   *
 592            ;*   - Instructions with select codes less than 20 are               *
 593            ;*     mapped a second time.                                         *
 594            ;*                                                                   *
 595            ;*   - All HLT instructions are mapped to same entry point           *
 596            ;*                                                                   *
 597            ;*********************************************************************
 598            ;*****
 599            ;*****    - All HLT's map here
 600            ;*****
 601            ;*****    - Algorithm is :  If Memory protect system is on,
 602            ;*****                        generate a memory protect interrupt
 603            ;*****                      Refetch instruction
 604            ;*****                      Wait for interrupt
 605            ;*****                      When an interrupt occurs, service it
 606            ;*****                        (hopefully, it is the slave request)
 607            ;*****
 608            HLT:      CJP   GENMPV     & AM2901 ,,NOP,ADD,AQ
 609   00045    /                          & CONDEXT MPEN    ; IS MP ON?
 610                      CONT             & AM2901 PC,PC,RAMF,SUBR,DA
 611            /                          & CARRYL          ;
 612            /                          & IMM H#0002      ;
 613   00046    /                          & LDMAR           ; REFETCH ADDRESS
 614                      CONT             & AM2901 ,PC,RAMF,ADD,ZB
 615            /                          & CARRYH          ;
 616   00047    /                          & RFETCH          ; PC -> HLT INST + 1
 617            HLT1:     CVECT INTTBL     & AM2901 ,,NOP,ADD,AQ
 618   00048    /                          & CONDEXT INTRPT ; (PROBABLY) SLAVE REQUEST
 619                      JP    HLT1       & AM2901 ,,NOP,ADD,AQ
 620   00049    /                          & SPNOP           ; WAIT FOR IT TO HAPPEN
```

```
LINE    ADDR    STATEMENT

 622            ;*************************************************************************
 623            ;*****                                                                   *
 624            ;*****    - Select Code >= 20Q I/O Instructions                         *
 625            ;*****    - Algorithm is :  If memory protect system is on, call        *
 626            ;*****        GENMPV                                                     *
 627            ;*****                          Refetch instruction                      *
 628            ;*****                          Set TDI, I/O handshake control word      *
 629            ;*****                          Call appropriate control word decode     *
 630            ;*****                            routine                                *
 631            ;*****                          Fetch next instruction                   *
 632            ;*****                                                                   *
 633            ;*************************************************************************
 634            ;*****
 635            IOGGE20:  CJP   GENMPV      & AM2901 PC,,NOP,ADD,AQ
 636            /                           & CONDEXT          ; CONDEXT IS MPEN (=SETTDI)
 637   0004A    /                           & ENCN SETTDI      ; GENMPV TURNS OFF TDI
 638            LT20ENT:  CONT              & AM2901 PC,,NOP,SUBR,DA
 639            /                           & CARRYL           ;
 640            /                           & IMM H#0002       ;
 641   0004B    /                           & LDMAR            ; LOAD MAR WITH REFETCH ADD
 642                      CALL  IOHSHAKE    & AM2901 ,,NOP,ADD,AQ
 643   0004C    /                           & RFETCH           ; GET I/O CONTROL WORD
 644                      CALL              & AM2901 ,,NOP,ADD,AQ
 645            /                           & SPETC IRSP       ; PUT CW ON IRBUS
 646   0004D    /                           & JTAB CWDCODE     ; DECODE I/O CONTROL WORD
 647            FIXMAR:   JZ                & AM2901 PC,,NOP,SUBR,ZA
 648            /                           & CARRYH           ;
 649            /                           & LDMAR            ;
 650   0004E    /                           & IFETCH           ; FETCH NEXT INSTRUCTION
```

```
LINE    ADDR    STATEMENT

652             ;*************************************************************************
653             ;*                                                                       *
654             ;*        Select Code < 20 I/O Instructions                              *
655             ;*                                                                       *
656             ;*  IOGLT20 - I/O GROUP, SC < 20 ENTRY POINT                             *
657             ;*                                                                       *
658             ;* -Basic Flow is to map instruction a second time and jump              *
659             ;*  to routine to perform function.                                      *
660             ;*                                                                       *
661             ;*  !! Note : ENCN SETTDI and CONDEXT MPEN must be identical             *
662             ;*            in bits PL12-PL15                                          *
663             ;*                                                                       *
664             ;*************************************************************************
665             ;*****
666             ;*****    - Select code < 20Q I/O routines
667             ;*****    - Algorithm is : Check if MP system on, and set TDI so
668             ;*****                       the next instruction gets executed
669             ;*****                     IF MP system off, load QREG with status
670             ;*****                       and jump to the appropriate routine
671             ;*****                       ELSE check if Select code is one.
672             ;*****                     IF select code = 1, execute the inst.,
673             ;*****                       ELSE generate a mem. prot. violation
674             ;*****
675             IOGLT20:  LDCT -          & AM2901 MP,MP,RAMF,SUBR,ZB
676             /                         & CARRYH          ; SUBTRACT 1 FROM SELECT CO
677             /                         & ENCN SETTDI     ; SET TEMP INT DISABLE
678   0004F     /                         & JTAB LOWSC      ; LDCT WITH ROUTINE ADDRESS
679                       JRP   MPLOWSC   & AM2901 ,,QREG,PASS,DZ
680             /                         & CONDEXT         ; CONDEXT IS MPEN (=STRD)
681   00050     /                         & SPRD STRD       ; READ STATUS REG
682             ;                                           ; MAPS INSTRUCTION IF MP SY
683             ;                                           ; IF MP SYS ON, CHECK FOR S
```

```
LINE    ADDR    STATEMENT

685             ;*************************************************************************
686             ;*                                                                       *
687             ;*        - Select Code 00 I/O Instructions                              *
688             ;*                (Interrupt system)                                     *
689             ;*                                                                       *
690             ;*************************************************************************
691             ;*****
692             ;*****    - CLC 00 - I/O System Reset (CRS-)
693             ;*****
694             ;*****    - Set STATUS reg as follows:
695             ;*****
696             ;*****       - TBG off
697             ;*****       - Int Inhibit off
698             ;*****       - Glogal Reg Flag disabled
699             ;*****       - Int Sys off
700             ;*****       - PS same
701             ;*****       - PONI same
702             ;*****
703             ;*****    - STATUS REGISTER :
704             ;*****
705             ;+------+------+------+------+------+------+------+------+------+------
706             ;|  9   |  8   |  7   |  6   |  5   |  4   |  3   |  2   |  1   |  0
707             ;+------+------+------+------+------+------+------+------+------+------
708             ;| PFW+ | TBT+ | GREN+| ISFF+| PONI+|TBGEN+| IIFF+| PS+  | TBG1+| TBG0+
709             ;+------+------+------+------+------+------+------+------+------+------
710             ;*****
711             ;*****    - Status bits 8 & 9 are read only (not updated by LDST)
712             ;*****
713             ;*****
714             CLC00:    JZ              & AM2901 ,,NOP,AND,DQ
715             /                         & LDST          ; LOAD STATUS REGISTER
716             /                         & ENCN ICRS     ; GENERATE CRS ON BACKPLANE
717             /                         & IMMB H#24     ; ZERO APPROPRIATE BITS
718   00051     /                         & IFETCH        ; FETCH NEXT INSTRUCTION
719             ;*****
720             ;*****    - CLF 00 - Disable Interrupt System
721             ;*****    - Note:  CLRF0 is used by SFS 00,C and SFC 00,C
722             ;*****
723             CLF00:    CONT            & AM2901 ,,NOP,ADD,AQ
724   00052     /                         & IFETCH        ;
725             CLRF0:    JZ              & AM2901 ,,NOP,NOTRS,DQ
726             /                         & LDST          ; LOAD STATUS REGISTER
727   00053     /                         & IMMB H#40     ; DISABLE TYPE 3 INTERRUPTS
728             ;
729             ;
730             LI.00C:   CONT            & AM2901 ,,NOP,NOTRS,DQ
731             /                         & LDST          ;
732   00054     /                         & IMMB H#40     ; CLEAR FLAG
733                       JP    LT20ENT   & AM2901 PC,,NOP,ADD,AQ
734   00055     /                         & SPNOP         ; I/O CHIP HANDSHAKES THE R
735             ;*****
736             ;*****    - OT. 00 - Output to interrupt mask register
737             ;*****    - Writes IMR bit 1, and then outputs A/B for the
```

```
LINE    ADDR    STATEMENT

 738            ;*****      I/O chips to examine.
 739            ;*****
 740            OT.00C:   CONT              & AM2901 ,,NOP,NOTRS,DQ
 741            /                           & LDST          ;
 742   00056    /                           & IMMB H#40     ; CLEAR FLAG
 743            OT.00:    JP    OT.02H      & AM2901 ,CAB,NOP,PASS,ZB
 744   00057    /                           & SPWR LDIM1    , LOAD TBG INTRPT MASK BIT
 745            ;*****
 746            ;*****    - SFC 00 - Skip if interrupt system disabled
 747            ;*****
 748            SFC00:    CONT              & AM2901 PC,PC,NOP,AND,DQ
 749            /                           & LUSRCOND      ;
 750   00058    /                           & IMM H#0040    ;
 751            SKIFZ:    JZ                & AM2901 PC,PC,RAMA,ADD,ZB
 752            /                           & CARRYEXT      ;
 753            /                           & CONDUSR Z     ;
 754            /                           & CLD & LDMAR   ;
 755   00059    /                           & IFETCH        ; SKIP IF TYPE 3 INTS DISAB
 756            ;*****
 757            ;*****    - SFC 00,C - Skip if interrupt system disabled and
 758            ;*****      disable interrupt system
 759            ;*****
 760            SFC00.C:  CONT              & AM2901 PC,PC,NOP,AND,DQ
 761            /                           & LUSRCOND      ;
 762   0005A    /                           & IMM H#0040    ; TEST BIT
 763                      JP    CLRF0       & AM2901 PC,PC,RAMA,ADD,ZB
 764            /                           & CARRYEXT      ;
 765            /                           & CONDUSR Z     ;
 766            /                           & CLD & LDMAR   ; SKIP IF TYPE 3 INTS DISAB
 767   0005B    /                           & IFETCH        ; DISABLE TYPE 3 INTERRUPTS
 768            ;*****
 769            ;*****    - SFS 00 - Skip if interrupt system enabled
 770            ;*****
 771            SFS00:    CONT              & AM2901 PC,PC,NOP,AND,DQ
 772            /                           & LUSRCOND      ;
 773   0005C    /                           & IMM H#0040    ;
 774            SKIFNZ:   JZ                & AM2901 PC,PC,RAMA,ADD,ZB
 775            /                           & CARRYEXT      ;
 776            /                           & CONDUSR NZ    ;
 777            /                           & CLD & LDMAR   ;
 778   0005D    /                           & IFETCH        ; SKIP IF TYPE 3 INTS ENABL
 779            ;*****
 780            ;*****    - SFS 00,C - Skip if interrupt system enabled and
 781            ;*****      disable interrupt system
 782            ;*****
 783            SFS00.C:  CONT              & AM2901 PC,PC,NOP,AND,DQ
 784            /                           & LUSRCOND      ;
 785   0005E    /                           & IMM H#0040    ;
 786                      JP    CLRF0       & AM2901 PC,PC,RAMA;ADD,ZB
 787            /                           & CARRYEXT      ;
 788            /                           & CONDUSR NZ    ;
 789            /                           & CLD & LDMAR   ; SKIP IF TYPE 3 INTS ENABL
 790   0005F    /                           & IFETCH        ; DISABLE TYPE 3 INTERRUPTS
 791            ;*****
```

```
LINE    ADDR    STATEMENT

 792            ;*****    - STF 00 - Enable Interrupt System
 793            ;*****
 794            STF00:    JZ              & AM2901 ,,NOP,OR,DQ
 795            /                         & LDST           ;
 796            /                         & IMMB H#40      ;
 797   00060    /                         & IFETCH         ; ENABLE TYPE 3 INTERRUPTS
```

```
LINE    ADDR    STATEMENT

 799            ;*********************************************************************
 800            ;*                                                                   *
 801            ;*        Select code 01 I/O instructions                            *
 802            ;*        (Overflow reg, LED reg, and switch reg)                    *
 803            ;*                                                                   *
 804            ;*********************************************************************
 805            CLO:      CONT              & AM2901 ,,NOP,ADD,AQ
 806   00061    /                           & IFETCH         ;
 807            CLRO:     JZ                & AM2901 ,,NOP,ADD,AQ
 808            /                           & LODMSR RESET   ;  CLEAR OVERFLOW
 809   00062    /                           & ENBLO          ;
 810            ;*****
 811            ;*****    - LI*01 - Read the SWITCH register
 812            ;*****    - OT*01 - Output to LED register
 813            ;*****    - The LED's on the A600+ are high true, which means that
 814            ;*****      a 1 written to a bit in the register will turn the
 815            ;*****      corresponding LED *ON*.
 816            ;*****
 817            LI.01C:   CONT              & AM2901 ,,NOP,ADD,AQ
 818            /                           & LODMSR RESET   ; CLEAR 0
 819   00063    /                           & ENBLO          ;
 820            LI.01:    CONT              & AM2901 ,CAB,RAMF,PASS,DZ
 821            /                           & SPRD SWRD      ;
 822   00064    /                           & IFETCH         ; INPUT FROM SWITCH REGISTE
 823                      JZ                & AM2901 CAB,CAB,RAMF,AND,DA
 824   00065    /                           & IMM H#FF00     ;
 825            MI.01C:   CONT              & AM2901 ,,NOP,ADD,AQ
 826            /                           & LODMSR RESET   ; CLEAR 0
 827   00066    /                           & ENBLO          ;
 828            MI.01:    CONT              & AM2901 ,S0,RAMF,PASS,DZ
 829   00067    /                           & SPRD SWRD      ;
 830                      CONT              & AM2901 S0,S0,RAMF,AND,DA
 831            /                           & IMM H#FF00     ;
 832   00068    /                           & IFETCH         ;
 833                      JZ                & AM2901 S0,CAB,RAMF,OR,AB
 834   00069    /                           & SPNOP          ;
 835            OT.01C:   JP    CLRO        & AM2901 ,CAB,NOP,EXNOR,ZB
 836            /                           & SPWR LEDWR     ;
 837   0006A    /                           & IFETCH         ; OUTPUT TO LED REG, CLEAR
 838            OT.01:    JZ                & AM2901 ,CAB,NOP,EXNOR,ZB
 839            /                           & SPWR LEDWR     ;
 840   0006B    /                           & IFETCH         ; OUTPUT TO LED REGISTER
 841            ;*****
 842            ;*****
 843            ;*****
 844            SOC.C:    JP    CLRO        & AM2901 PC,PC,RAMA,ADD,ZB
 845            /                           & CARRYEXT       ;
 846            /                           & CONDMSR NOVR   ;
 847            /                           & CLD & LDMAR    ;
 848   0006C    /                           & IFETCH         ;
 849            SOC.H:    JZ                & AM2901 PC,PC,RAMA,ADD,ZB
 850            /                           & CARRYEXT       ;
 851            /                           & CONDMSR NOVR   ;
```

```
LINE    ADDR    STATEMENT

 852            /                       & CLD & LDMAR     ;
 853    0006D   /                       & IFETCH          ; SKIP IF OVERFLOW CLEAR
 854            SOS.C:    JP    CLRO    & AM2901 PC,PC,RAMA,ADD,ZB
 855            /                       & CARRYEXT        ;
 856            /                       & CONDMSR OVR     ;
 857            /                       & CLD & LDMAR     ;
 858    0006E   /                       & IFETCH          ;
 859            SOS.H:    JZ            & AM2901 PC,PC,RAMA,ADD,ZB
 860            /                       & CARRYEXT        ;
 861            /                       & CONDMSR OVR     ;
 862            /                       & CLD & LDMAR     ;
 863    0006F   /                       & IFETCH          ; SKIP IF OVERFLOW SET
 864            STO:      JZ            & AM2901 ,,NOP,ADD,AQ
 865            /                       & IFETCH          ;
 866            /                       & LODMSR SET      ; SET OVERFLOW
 867    00070   /                       & ENBLO           ;
```

```
LINE    ADDR    STATEMENT

 869            ;*********************************************************************
 870            ;*                                                                   *
 871            ;*        Select Code 02 I/O Instructions                            *
 872            ;*        (Global reg and enable BREAK)                              *
 873            ;*                                                                   *
 874            ;*********************************************************************
 875            ;*****
 876            ;*****    - CLF 02 - Enable global register
 877            ;*****
 878            CLF02:    CONT              & AM2901 PC,,NOP,SUBR,DA
 879            /                           & CARRYL          ;
 880            /                           & IMM H#0002      ;
 881   00071    /                           & LDMAR           ;
 882                      CONT              & AM2901 PC,,NOP,OR,DQ
 883            /                           & LDST            ;
 884            /                           & IMMB H#80       ;
 885   00072    /                           & RFETCH          ; SET GR BIT, REFETCH
 886                      JZ                & AM2901 PC,,NOP,SUBR,ZA
 887            /                           & CARRYH          ;
 888            /                           & LDMAR           ;
 889   00073    /                           & IFETCH          ; FETCH NEXT INSTRUCTION
 890            ;
 891            LI.02C:   CONT              & AM2901 ,,NOP,OR,DQ
 892            /                           & LDST            ;
 893   00074    /                           & IMMB H#80       ; SET GR BIT
 894                      JP     LT20ENT    & AM2901 PC,,NOP,ADD,AQ
 895   00075    /                           & SPNOP           ;
 896            ;*****
 897            ;*****    - LI* 02  MAPS TO LT20ENT
 898            ;*****
 899            ;
 900            ;*****
 901            ;*****    - OT* 02 - Output to global register
 902            ;*****
 903            ;*****    - OT.02H - Output to global register
 904            ;*****    - OT.02C - Output to global register and clear flag
 905            ;*****                 (enable the GR)
 906            ;*****             - Sets GREN+ flag and outputs A/B for the I/O
 907            ;*****                 chips to examine.
 908            ;*****
 909            OT.02C:   CONT              & AM2901 PC,,NOP,OR,DQ
 910            /                           & LDST            ;
 911   00076    /                           & IMMB H#80       ;
 912            OT.02H:   CONT              & AM2901 PC,,NOP,SUBR,DA
 913            /                           & CARRYL          ;
 914            /                           & IMM H#0002      ;
 915   00077    /                           & LDMAR           ;
 916                      CALL   IOHSHAKE   & AM2901 ,,NOP,ADD,AQ
 917   00078    /                           & RFETCH          ;
 918                      CALL   IOHSHAK2   & AM2901 ,CAB,NOP,PASS,ZB
 919            /                           & LDMDOR          ;
 920   00079    /                           & SPETC IOWR      ; PUT A/B ON DBUS FOR I/O
 921                      JZ                & AM2901 PC,,NOP,SUBR,ZA
```

```
LINE   ADDR   STATEMENT

 922          /                          & CARRYH          ;
 923          /                          & LDMAR           ;
 924   0007A  /                          & IFETCH          ;
 925          ;*****
 926          ;*****   - SFC 02 - Skip if global register enabled
 927          ;*****
 928          SFC02C:  CALL  PAT02C      & AM2901 ,,NOP,ADD,AQ
 929   0007B  /                          & SPNOP           ;
 930                   JP    CLF02       & AM2901 PC,PC,RAMF,ADD,ZB
 931          /                          & CARRYEXT        ;
 932   0007C  /                          & CONDUSR NZ      ; IN PC IF SKIP
 933          ;
 934          SFC02:   CONT              & AM2901 PC,PC,NOP,AND,DQ
 935          /                          & LUSRCOND        ;
 936   0007D  /                          & IMM H#0080      ; TEST GLOBAL REG BIT
 937                   JZ                & AM2901 PC,PC,RAMA,ADD,ZB
 938          /                          & CARRYEXT        ;
 939          /                          & CONDUSR NZ      ;
 940          /                          & CLD & LDMAR     ;
 941   0007E  /                          & IFETCH          ; SKIP IF SET
 942          ;*****
 943          ;*****   - SFS 02 - Skip if global register disabled
 944          ;*****
 945          SFS02C:  CALL  PAT02C      & AM2901 ,,NOP,ADD,AQ
 946   0007F  /                          & SPNOP           ;
 947                   JP    CLF02       & AM2901 PC,PC,RAMF,ADD,ZB
 948          /                          & CARRYEXT        ;
 949   00080  /                          & CONDUSR Z       ; INC PC IF SKIP
 950          ;
 951          SFS02:   CONT              & AM2901 PC,PC,NOP,AND,DQ
 952          /                          & LUSRCOND        ;
 953   00081  /                          & IMM H#0080      ; TEST GLOBAL REG BIT
 954                   JZ                & AM2901 PC,PC,RAMA,ADD,ZB
 955          /                          & CARRYEXT        ;
 956          /                          & CONDUSR Z       ;
 957          /                          & CLD & LDMAR     ;
 958   00082  /                          & IFETCH          ; SKIP IF CLEAR
 959          ;*****
 960          ;*****   - STC 02 - Enable break feature  (NOT like L-SERIES)
 961          ;*****
 962          STC02:   CONT              & AM2901 PC,,NOP,SUBR,DA
 963          /                          & CARRYL          ;
 964          /                          & IMM H#0002      ;
 965   00083  /                          & LDMAR           ; MAR := REFETCH ADDRESS
 966                   JP    FIXMAR      & AM2901 PC,,NOP,ADD,AQ
 967   00084  /                          & RFETCH          ; REFETCH THE STC02 ;
 968          ;*****                                        GO FETCH NEXT INSTRUCTION
 969          ;*****   - STF 02 - Disable Global register
 970          ;*****   - NOTE : Global register enable bit is reverse sense
 971          ;*****
 972          STF02:   CONT              & AM2901 PC,,NOP,SUBR,DA
 973          /                          & CARRYL          ;
 974          /                          & IMM H#0002      ;
 975   00085  /                          & LDMAR           ; MAR := REFETCH ADDRESS
```

```
LINE    ADDR    STATEMENT

 976                    CONT            & AM2901 PC,,NOP,NOTRS,DQ
 977            /                       & LDST          ;
 978            /                       & RFETCH        ; REGETCH INSTRUCTION
 979   00086    /                       & IMMB H#80     ; CLEAR GLOBAL REG BIT
 980                    JZ              & AM2901 PC,,NOP,SUBR,ZA
 981            /                       & CARRYH        ;
 982            /                       & LDMAR         ;
 983   00087    /                       & IFETCH        ; FETCH NEXT INSTRUCTION
```

```
LINE    ADDR    STATEMENT

 985            ;*********************************************************************
 986            ;*                                                                   *
 987            ;*        Select Code 04 I/O Instructions                            *
 988            ;*        (Type 2 & 3 ints, PFW, and CIR)                            *
 989            ;*                                                                   *
 990            ;*********************************************************************
 991            CLC04:    JZ              & AM2901 ,,NOP,OR,DQ
 992            /                         & LDST           ;
 993            /                         & IMM H#0008     ;
 994   00088    /                         & IFETCH         ; DISABLE TYPE 2 AND 3 INTE
 995            ;
 996            LI.04:    JZ              & AM2901 CIR,CAB,RAMF,PASS,DZ
 997   00089    /                         & IFETCH         ; CIR TO A/B
 998            ;
 999            MI.04:    CONT            & AM2901 CIR,S0,RAMF,PASS,DZ
1000   0008A    /                         & IFETCH         ; S0 = CIR
1001                      JZ              & AM2901 S0,CAB,RAMF,OR,AB
1002   0008B    /                         & SPNOP          ; MERGE CIR
1003            ;
1004            OT.04:    JZ              & AM2901 CAB,CIR,NOP,AND,DA
1005            /                         & IMMB H#3F      ;
1006   0008C    /                         & IFETCH         ; A/B TO CIR (6 BITS ONLY)
1007            ;*****
1008            ;*****    - SFC 04 - Skip if power going down
1009            ;*****    - NOTE : Status bit is reverse sense
1010            ;*****
1011            SFC04:    CONT            & AM2901 PC,PC,NOP,AND,DQ
1012            /                         & LUSRCOND       ;
1013   0008D    /                         & IMM H#0200     ; PFW BIT
1014                      JZ              & AM2901 PC,PC,RAMA,ADD,ZB
1015            /                         & CARRYEXT       ;
1016            /                         & CONDUSR NZ     ;
1017            /                         & CLD & LDMAR    ;
1018   0008E    /                         & IFETCH         ; SKIP IF SET
1019            ;*****
1020            ;*****    - SFS 04 - Skip if power not going down
1021            ;*****    - NOTE : Status bit is reverse sense
1022            ;*****
1023            SFS04:    CONT            & AM2901 PC,PC,NOP,AND,DQ
1024            /                         & LUSRCOND       ;
1025   0008F    /                         & IMM H#0200     ; PONI BIT
1026                      JZ              & AM2901 PC,PC,RAMA,ADD,ZB
1027            /                         & CARRYEXT       ;
1028            /                         & CONDUSR Z      ;
1029            /                         & CLD & LDMAR    ;
1030   00090    /                         & IFETCH         ; SKIP IF CLEAR
1031            STC04:    JZ              & AM2901 ,,NOP,NOTRS,DQ
1032            /                         & LDST           ;
1033            /                         & IMMB H#08      ;
1034   00091    /                         & IFETCH         ; ENABLE TYPE 2 AND 3 INTER
```

```
LINE    ADDR    STATEMENT

1036            ;*********************************************************************
1037            ;*                                                                   *
1038            ;*        Select Code 05 I/O Instructions                            *
1039            ;*        (Parity system)                                            *
1040            ;*                                                                   *
1041            ;*********************************************************************
1042            CLC05C:   JZ              & AM2901 ,,NOP,NOTRS,DQ
1043            /                         & LDST           ;
1044            /                         & ENCN CLRPSFF   ;
1045            /                         & IMMB H#04      ;
1046    00092   /                         & IFETCH         ; DISABLE PARITY, SET ODD
1047            CLC05:    JZ              & AM2901 ,,NOP,ADD,AQ
1048            /                         & ENCN CLRPSFF   ;
1049    00093   /                         & IFETCH         ; DISABLE PARITY SYSTEM
1050            CLF05:    CONT            & AM2901 ,,NOP,ADD,AQ
1051    00094   /                         & IFETCH         ; FETCH NEXT INST
1052            CLRF5:    JZ              & AM2901 ,,NOP,NOTRS,DQ
1053            /                         & LDST           ;
1054    00095   /                         & IMMB H#04      ; SET ODD PARITY SENSE
1055            LI.05H:   CONT            & AM2901 ,CAB,RAMF,PASS,DZ
1056            /                         & SPRD PELENL    ;
1057    00096   /                         & IFETCH         ; LOAD PARITY LOW ORDER REG
1058                      JZ              & AM2901 CAB,CAB,RAMF,EXOR,DA
1059    00097   /                         & IMM H#FC00     ; INVERT HIGH 6 BITS
1060            LI.05C:   CONT            & AM2901 ,CAB,RAMF,PASS,DZ
1061    00098   /                         & SPRD PELENH    ; LOAD PARITY HIGH ORDER RE
1062                      CONT            & AM2901 CAB,,NOP,AND,DA
1063            /                         & IMM H#FF00     ; MASK OFF LOW BYTE
1064    00099   /                         & LUSRCOND       ;
1065                      CJP   LI05A     & AM2901 ,,NOP,ADD,AQ
1066            /                         & CONDUSR NZ     ;
1067    0009A   /                         & IFETCH         ;
1068                      JZ              & AM2901 CAB,CAB,RAMF,EXNOR,DA
1069    0009B   /                         & IMM H#FF00     ; INVERT LOW BYTE ONLY
1070            LI05A:    JZ              & AM2901 CAB,CAB,RAMF,EXNOR,DA
1071    0009C   /                         & IMM H#C000     ; INVERT ALL BUT TOP TWO BI
1072            SFC05C:   CONT            & AM2901 PC,PC,NOP,AND,DQ
1073            /                         & LUSRCOND       ;
1074    0009D   /                         & IMM H#0004     ;
1075                      JP    CLRF5     & AM2901 PC,PC,RAMA,ADD,ZB
1076            /                         & CARRYEXT       ;
1077            /                         & CONDUSR Z      ;
1078            /                         & CLD & LDMAR    ;
1079    0009E   /                         & IFETCH         ;
1080            SFC05:    CONT            & AM2901 PC,PC,NOP,AND,DQ
1081            /                         & LUSRCOND       ;
1082    0009F   /                         & IMM H#0004     ; SKIP IF PS ODD
1083                      JZ              & AM2901 PC,PC,RAMA,ADD,ZB
1084            /                         & CARRYEXT       ;
1085            /                         & CONDUSR Z      ;
1086            /                         & CLD & LDMAR    ;
1087    000A0   /                         & IFETCH         ; SKIP IF CLEAR
1088            SFS05C:   CONT            & AM2901 PC,PC,NOP,AND,DQ
```

```
LINE    ADDR    STATEMENT

1089            /                       & LUSRCOND        ;
1090    000A1   /                       & IMM H#0004      ;
1091                    JP      CLRF5   & AM2901 PC,PC,RAMA,ADD,ZB
1092            /                       & CARRYEXT        ;
1093            /                       & CONDUSR NZ      ;
1094            /                       & CLD & LDMAR     ;
1095    000A2   /                       & IFETCH          ;
1096            SFS05:  CONT            & AM2901 PC,PC,NOP,AND,DQ
1097            /                       & LUSRCOND        ;
1098    000A3   /                       & IMM H#0004      ; SKIP IF PS EVEN
1099                    JZ              & AM2901 PC,PC,RAMA,ADD,ZB
1100            /                       & CARRYEXT        ;
1101            /                       & CONDUSR NZ      ;
1102            /                       & CLD & LDMAR     ;
1103    000A4   /                       & IFETCH          ; SKIP IF SET
1104            STC05C: JZ              & AM2901 ,,NOP,NOTRS,DQ
1105            /                       & LDST            ;
1106            /                       & ENCN SETPSFF    ;
1107            /                       & IMMB H#04       ;
1108    000A5   /                       & IFETCH          ; ENABLE PARITY, SET ODD PA
1109            STC05:  JZ              & AM2901 ,,NOP,ADD,AQ
1110            /                       & ENCN SETPSFF    ;
1111    000A6   /                       & IFETCH          ; ENABLE PARITY SYSTEM
1112            STF05:  JZ              & AM2901 ,,NOP,OR,DQ
1113            /                       & LDST            ;
1114            /                       & IMMB H#04       ;
1115    000A7   /                       & IFETCH          ; SET EVEN PARITY SENSE (1)
```

```
LINE    ADDR    STATEMENT

1117            ;*********************************************************************
1118            ;*                                                                   *
1119            ;*       Select Code 06 I/O Instructions                             *
1120            ;*       (TBG)                                                       *
1121            ;*                                                                   *
1122            ;*********************************************************************
1123            ;*****
1124            ;*****   - CLC 06   - Turn off TBG (same as CLC 06,C)
1125            ;*****   - CLC 06,C - Turn off TBG and clear flag
1126            ;*****
1127            CLC06:  JZ              & AM2901 ,,NOP,NOTRS,DQ
1128            /                       & ENCN CLRTBT    ;
1129            /                       & IMMB H#10      ;
1130            /                       & LDST           ; CLEAR TBG ENABLE BIT
1131    000A8   /                       & IFETCH         ; AND TBG TICK FLAG
1132            ;*****
1133            ;*****   - CLF 06 - Clear flag on TBG
1134            ;*****
1135            CLF06:  CONT            & AM2901 ,,NOP,ADD,AQ
1136    000A9   /                       & IFETCH         ; FETCH NEXT INST
1137            CLRF6:  JZ              & AM2901 ,,NOP,ADD,AQ
1138    000AA   /                       & ENCN CLRTBT    ; CLEAR TBG TIC FLAG
1139            ;*****
1140            ;*****   - LI* 06 - Read TBG frequency register
1141            ;*****
1142            LI.06C: CONT            & AM2901 ,,NOP,ADD,AQ
1143    000AB   /                       & ENCN CLRTBT    ; CLEAR TIC
1144            LI.06:  CONT            & AM2901 ,S0,RAMF,SUBS,ZQ
1145            /                       & CARRYH         ;
1146    000AC   /                       & IFETCH         ; S0 = PROPER BITS; FETCH
1147                    JZ              & AM2901 S0,CAB,RAMF,AND,DA
1148    000AD   /                       & IMM H#0003     ; MASK TO 2 BITS
1149            ;*****
1150            ;*****   - OT* 06 - Write TBG frequency register
1151            ;*****
1152            OT.06C: CONT            & AM2901 ,,NOP,ADD,AQ
1153    000AE   /                       & ENCN CLRTBT    ; CLEAR TIC
1154            OT.06:  CONT            & AM2901 CAB,S0,RAMF,SUBS,ZA
1155    000AF   /                       & CARRYH         ; S0 = PROPER BITS
1156                    CONT            & AM2901 S0,S0,RAMF,AND,DA
1157    000B0   /                       & IMM H#0003     ; MASK TO TWO BITS
1158                    CONT            & AM2901 ,,QREG,NOTRS,DQ
1159            /                       & IMM H#0003     ;
1160    000B1   /                       & IFETCH         ; CLEAR TWO BITS IN Q, FETC
1161                    JZ              & AM2901 S0,,NOP,OR,AQ
1162    000B2   /                       & LDST           ; OR IN TWO NEW BITS
1163            ;*****
1164            ;*****   - SFC 06 - Skip if TBG flag clear
1165            ;*****
1166            SFC06C: CONT            & AM2901 PC,PC,NOP,AND,DQ
1167            /                       & LUSRCOND       ;
1168    000B3   /                       & IMM H#0100     ; TBG BIT
1169                    JP    CLRF6     & AM2901 PC,PC,RAMA,ADD,ZB
```

```
LINE    ADDR    STATEMENT

1170            /                       & CARRYEXT      ;
1171            /                       & CONDUSR Z     ;
1172            /                       & CLD & LDMAR   ;
1173    000B4   /                       & IFETCH        ; SKIP IF CLEAR
1174            ;
1175            SFC06:  CONT            & AM2901 PC,PC,NOP,AND,DQ
1176            /                       & LUSRCOND      ;
1177    000B5   /                       & IMM H#0100    ; TBG BIT
1178                    JZ              & AM2901 PC,PC,RAMA,ADD,ZB
1179            /                       & CARRYEXT      ;
1180            /                       & CONDUSR Z     ;
1181            /                       & CLD & LDMAR   ;
1182    000B6   /                       & IFETCH        ; SKIP IF CLEAR
1183            ;*****
1184            ;*****  - SFS 06 - Skip if TBG flag set
1185            ;*****
1186            SFS06C: CONT            & AM2901 PC,PC,NOP,AND,DQ
1187            /                       & LUSRCOND      ;
1188    000B7   /                       & IMM H#0100    ; TEST TBG BIT
1189                    JP    CLRF6     & AM2901 PC,PC,RAMA,ADD,ZB
1190            /                       & CARRYEXT      ;
1191            /                       & CONDUSR NZ    ;
1192            /                       & CLD & LDMAR   ;
1193    000B8   /                       & IFETCH        ; SKIP IF SET
1194            ;
1195            SFS06:  CONT            & AM2901 PC,PC,NOP,AND,DQ
1196            /                       & LUSRCOND      ;
1197    000B9   /                       & IMM H#0100    ; TBG BIT
1198                    JZ              & AM2901 PC,PC,RAMA,ADD,ZB
1199            /                       & CARRYEXT      ;
1200            /                       & CONDUSR NZ    ;
1201            /                       & CLD & LDMAR   ;
1202    000BA   /                       & IFETCH        ; SKIP IF SET
1203            ;*****
1204            ;*****  - STC 06 - Turn on TBG
1205            ;*****
1206            STC06C: JZ              & AM2901 ,,NOP,OR,DQ
1207            /                       & LDST          ;
1208            /                       & ENCN CLRTBT   ;
1209            /                       & IMMB H#10     ;
1210    000BB   /                       & IFETCH        ; SET TBG ENABLE, CLEAR FLA
1211            STC06:  JZ              & AM2901 ,,NOP,OR,DQ
1212            /                       & LDST          ;
1213            /                       & IMMB H#10     ;
1214    000BC   /                       & IFETCH        ; SET TBG ENABLE
1215            ;*****
1216            ;*****  - STF 06 - Set flag on TBG
1217            ;*****
1218            STF06:  JZ              & AM2901 ,,NOP,ADD,AQ
1219            /                       & ENCN SETTBT   ;
1220    000BD   /                       & IFETCH        ; SET TBG TICK FLAG
```

```
LINE    ADDR    STATEMENT

1222            ;*****************************************************************
1223            ;*                                                               *
1224            ;*        Select Code 07 I/O Instructions                        *
1225            ;*        (Memory protect)                                       *
1226            ;*                                                               *
1227            ;*****************************************************************
1228            ;*****
1229            ;*****    - LI. 07 - Input from Mem Prot Violation Reg
1230            ;*****
1231            LI.07:    JZ              & AM2901 ,CAB,RAMF,PASS,DZ
1232            /                         & SPRD PRLEN      ;
1233   000BE    /                         & IFETCH          ; LOAD FROM VIOLATION REG
1234            ;*****
1235            ;*****    - STC 07 - Turn on Memory Protect
1236            ;*****
1237            STC07:    JZ              & AM2901 ,,NOP,PASS,DZ
1238            /                         & ENCN SETMPEN    ;
1239   000BF    /                         & IFETCH          ; TURN ON MEMORY PROT.
```

```
LINE    ADDR    STATEMENT

1241            ;*********************************************************************
1242            ;*                                                                   *
1243            ;*       Extended Arithmetic Register Group                          *
1244            ;*       ----------------------------------                          *
1245            ;*                                                                   *
1246            ;*********************************************************************
1247            ;*****
1248            ;*****   - Arithmetic shift left
1249            ;*****
1250            ;*****
1251            ;*****   - Arithmetic shift left
1252            ;*****   - Algorithm is :  Clear O
1253            ;*****                     Swap E & O
1254            ;*****                     Set QREG := A
1255            ;*****                     2910 := shift count from IR
1256            ;*****                       1: Double shift B and QREG
1257            ;*****                          Check for sign change, loop to 1:
1258            ;*****                     Restore sign
1259            ;*****                     A := QREG
1260            ;*****                     Swap E & O back
1261            ;*****
1262            ;*****                     If shifting caused a sign change, E
1263            ;*****                       is set, which becomes an overflow
1264            ;*****                       when E and O are swapped.
1265            ;*****
1266            ASL:    CONT            & AM2901 ,A,QREG,PASS,ZB
1267            /                       & LODMSR RESET   ;
1268    000C0   /                       & ENBLO          ; RESET OVERFLOW
1269                    PUSH            & AM2901 ,B,SRAMQL,PASS,ZB
1270            /                       & LODMSR SWAPEO  ;
1271            /                       & ENBLC & ENBLO  ; SET MSR SGN = B
1272            /                       & JTAB IR0T3     ; LOAD 2910 COUNTER
1273    000C1   /                       & SHIFT B#0110   ;
1274                    CCALL ASLOVFL   & AM2901 ,B,NOP,PASS,ZB
1275    000C2   /                       & DIVCOND        ; IF SIGN EXOR MN NO MATCH
1276                    RFCT            & AM2901 ,B,SRAMQL,PASS,ZB
1277    000C3   /                       & SHIFT B#0110   ;
1278                    CONT            & AM2901 ,B,SRAMQR,PASS,ZB
1279            /                       & SHIFT B#0101   ; RESTORE SIGN
1280    000C4   /                       & IFETCH         ; FETCH NEXT INSTRUCTION
1281                    JZ              & AM2901 ,A,RAMF,PASS,ZQ
1282            /                       & LODMSR SWAPEO  ; FINISH UP
1283    000C5   /                       & ENBLC & ENBLO  ; PUT A REG, E REG BACK
1284    000C6           FILLER
1285    000C7           FILLER
1286            ;*****
1287            ;*****  - Jump and load A/B
1288            ;*****
1289            JL.:    CCALL INDRES1   & AM2901 BR,TABQ,QREG,ADD,DZ
1290            /                       & CARRYL         ;
1291            /                       & CONDEXT MDIR15 ;
1292            /                       & LDMAR          ;
1293    000C8   /                       & CMGO & DREAD   ; RESOLVE INDIRECT
```

```
LINE   ADDR   STATEMENT

1294                  JP    INCFTCH    & AM2901 PC,CAB,RAMF,PASS,ZA
1295   000C9  /                         & SPNOP          ; LOAD A/B WITH OLD PC & FE
1296          ;*****
1297          ;*****  - Rotate left
1298          ;*****  - Algorithm is :  QREG := A
1299          ;*****                    Load 2910 counter with shift count
1300          ;*****                      from IR
1301          ;*****                  1: Double rotate B and QREG, loop to 1:
1302          ;*****                     A := QREG
1303          ;*****
1304          RRL:    PUSH              & AM2901 ,A,QREG,PASS,ZB
1305          /                         & LODMSR RESET   ;
1306          /                         & JTAB IR0T3     ;
1307   000CA  /                         & IFETCH         ;
1308                  TWB   JZA         & AM2901 ,B,SRAMQL,PASS,ZB
1309          /                         & CONDMSR SGN    ; NEVER FALL THROUGH
1310   000CB  /                         & SHIFT B#1111   ;
1311          ;*****
1312          ;*****  - Rotate Right
1313          ;*****  - Algorithm is :  QREG := A
1314          ;*****                    Load 2910 counter with shift count
1315          ;*****                      from IR
1316          ;*****                  1: Double rotate B and QREG, loop to 1:
1317          ;*****                     A := QREG
1318          ;*****
1319          RRR:    CONT              & AM2901 ,S0,RAMF,PASS,DZ
1320   000CC  /                         & IMM H#000F     ; S0 = MASK
1321                  CONT              & AM2901 S0,,NOP,AND,DA
1322   000CD  /                         & LODMSR         ; MSR = SHIFT COUNT
1323                  PUSH              & AM2901 ,A,QREG,PASS,ZB
1324          /                         & JTAB IR0T3     ;
1325   000CE  /                         & IFETCH         ;
1326                  TWB   JZA         & AM2901 ,B,SRAMQR,PASS,ZB
1327   000CF  /                         & CONDMSR Z      ; SHIFT IS B#1111
1328                  CONT              & AM2901 B,A,SRAMQL,PASS,ZA
1329   000D0  /                         & SHIFT B#1111   ; SWAP A & B
1330                  JZ                & AM2901 ,B,RAMF,PASS,ZQ
1331   000D1  /                         & SPNOP          ;
1332          ;*****
1333          ;*****  - Arithmetic shift right
1334          ;*****  - Algorithm is :  Load MSR with sign
1335          ;*****                    Load QREG with A
1336          ;*****                    Load 2910 counter with shift count
1337          ;*****                      from IR
1338          ;*****                  1: Double shift B and QREG, loop to 1:
1339          ;*****                     A := QREG
1340          ;*****                     Clear 0
1341          ;*****
1342          ASR:    CONT              & AM2901 ,B,NOP,PASS,ZB
1343   000D2  /                         & LODMSR         ; SET SIGN
1344                  PUSH              & AM2901 ,A,QREG,PASS,ZB
1345          /                         & JTAB IR0T3     ;
1346   000D3  /                         & IFETCH         ;
1347                  RFCT              & AM2901 ,B,SRAMQR,PASS,ZB
```

```
LINE   ADDR   STATEMENT

1348   000D4  /                        & SHIFT B#0101    ;
1349                 JZ                & AM2901 ,A,RAMF,PASS,ZQ
1350          /                        & LODMSR RESET    ; RESTORE A
1351   000D5  /                        & ENBLO           ; CLEAR OVERFLOW
1352          ;*****
1353          ;*****  - Logical shift left
1354          ;*****  - Algorithm is :  QREG := A
1355          ;*****                    2910 counter loaded with shift count
1356          ;*****                      from IR
1357          ;*****                  1: Double shift B and QREG, loop to 1:
1358          ;*****                     A := QREG
1359          ;*****
1360          LSL:    PUSH              & AM2901 ,A,QREG,PASS,ZB
1361          /                         & JTAB IR0T3      ;
1362   000D6  /                         & IFETCH          ;
1363                 RFCT               & AM2901 ,B,SRAMQL,PASS,ZB
1364   000D7  /                         & SHIFT B#0110    ;
1365                 JZ                 & AM2901 ,A,RAMF,PASS,ZQ
1366   000D8  /                         & SPNOP           ; RESTORE A
1367          ;*****
1368          ;*****  - Logical shift right
1369          ;*****  - Algorithm is :  QREG := A
1370          ;*****                    Load 2910 counter with shift count
1371          ;*****                      from IR
1372          ;*****                  1: Double shift B and QREG, loop to 1:
1373          ;*****                     A := QREG
1374          ;*****
1375          LSR:    PUSH              & AM2901 ,A,QREG,PASS,ZB
1376          /                         & JTAB IR0T3      ;
1377   000D9  /                         & IFETCH          ;
1378                 RFCT               & AM2901 ,B,SRAMQR,PASS,ZB
1379   000DA  /                         & SHIFT B#0110    ;
1380                 JZ                 & AM2901 ,A,RAMF,PASS,ZQ
1381   000DB  /                         & SPNOP           ; RESTORE A
```

```
LINE    ADDR    STATEMENT

1383            ;*************************************************************************
1384            ;*                                                                       *
1385            ;*        Extended Arithmetic Group                                      *
1386            ;*        -------------------------                                      *
1387            ;*                                                                       *
1388            ;*        opcode word                                                    *
1389            ;*        D/I | addr word                                                *
1390            ;*                                                                       *
1391            ;*************************************************************************
1392            ;*****
1393            ;*************************************************************************
1394            ;*                                                                       *
1395            ;*        DIVIDE - 1ST AND 2ND QUADRANT SIGNED DIVISOR                   *
1396            ;*                                                                       *
1397            ;*     - Dividend is in macro B and A registers                          *
1398            ;*     - Dividend is made positive                                       *
1399            ;*     - Divisor is a two's complement signed integer                    *
1400            ;*                                                                       *
1401            ;*     - At each iteration step, the partial remainder is                *
1402            ;*       reduced toward zero by add or subtract : If partial             *
1403            ;*       remainder sign XOR divisor sign = 0 then SUBR, ELSE ADD         *
1404            ;*                                                                       *
1405            ;*     - Remainder sign is always same as dividend sign                  *
1406            ;*     - Quotient sign is dividend sign XOR divisor sign                 *
1407            ;*                                                                       *
1408            ;*     - Overflow if divide by zero leaves ABS(dividend) in B&A          *
1409            ;*     - Overflow if divisor too small leaves B & A undefined            *
1410            ;*     - Quotient bits are generated in one's complement form            *
1411            ;*                                                                       *
1412            ;*************************************************************************
1413            ;*****
1414            ;*****    - Get divisor
1415            ;*****
1416            DIVD:     CCALL READRES    & AM2901 BR,TABQ,QREG,ADD,DZ
1417            /                          & CARRYL         ;
1418            /                          & CONDEXT MDIR15 ;
1419            /                          & LDMAR          ;
1420   000DC    /                          & DREAD          ; RESOLVE DEF
1421            ;*****
1422            ;*****    - Save sign of dividend in S1
1423            ;*****
1424                      JP    DIVCONT    & AM2901 B,S1,RAMF,PASS,ZA
1425   000DD    /                          & LODUSR         ; GO TO DIVIDE ROUTINE
```

```
LINE    ADDR    STATEMENT

1427            ;*************************************************************************
1428            ;*****                                                                   *
1429            ;*****    - Double load A and B from MEM and MEM+1                      *
1430            ;*****                                                                   *
1431            ;*************************************************************************
1432            DLD:      CCALL READRES    & AM2901 BR,TABQ,QREG,ADD,DZ
1433            /                          & CARRYL         ;
1434            /                          & CONDEXT MDIR15 ;
1435            /                          & LDMAR          ;
1436   000DE    /                          & DREAD          ; READ FIRST WORD
1437                      CONT             & AM2901 ,,QREG,ADD,ZQ
1438            /                          & CARRYH         ;
1439   000DF    /                          & LDMAR          ; MAR = SECOND WORD
1440                      CALL  FIXPC      & AM2901 TAB,A,RAMF,PASS,DZ
1441   000E0    /                          & DREAD          ; LOAD A WITH FIRST WORD
1442                      JZ               & AM2901 TAB,B,RAMF,PASS,DZ
1443   000E1    /                          & IFETCH         ; LOAD B WITH SECOND WORD,
1444            ;
1445            ;*************************************************************************
1446            ;*****                                                                   *
1447            ;*****    - Double store A & B to MEM and MEM+1                         *
1448            ;*****                                                                   *
1449            ;*************************************************************************
1450            DST:      CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
1451            /                          & CARRYL         ;
1452            /                          & CONDEXT MDIR15 ;
1453            /                          & LDMAR          ;
1454   000E2    /                          & CMG0 & DREAD   ;  IF INDIRECT
1455                      CALL  INCQ       & AM2901 A,TAB,NOP,PASS,ZA
1456            /                          & LDMDOR         ;
1457   000E3    /                          & DWRITE         ; STORE A
1458                      JP    SKIP       & AM2901 B,TAB,NOP,PASS,ZA
1459            /                          & LDMDOR         ;
1460   000E4    /                          & DWRITE         ; STORE B
```

```
LINE    ADDR    STATEMENT

1462            ;*********************************************************************
1463            ;*                                                                   *
1464            ;*        MULTIPLY - 2'S COMPLEMENT SIGNED                           *
1465            ;*                                                                   *
1466            ;*********************************************************************
1467            ;*****
1468            MPY0:     CCALL READRES    & AM2901 BR,TABQ,QREG,ADD,DZ
1469            /                          & CARRYL          ;
1470            /                          & CONDEXT MDIR15  ;
1471            /                          & LDMAR           ;
1472   000E5    /                          & DREAD           ; FETCH ADDR WORD
1473                      CONT             & AM2901 PC,PC,RAMA,ADD,ZB
1474            /                          & CARRYH          ;
1475   000E6    /                          & LDMAR           ; MAR = NEXT INST, INC PC
1476                      CONT             & AM2901 TAB,,QREG,PASS,DZ
1477   000E7    /                          & SPNOP           ; MULTIPLIER TO QREG
1478            ;*****
1479            ;*****    - Zero B Reg and shift Q right one bit.  This
1480            ;*****        puts Q0 into Q0BUF flip-flop for multiply step.
1481            ;*****
1482            ;*****   Notes: - In next line, CARRYL is used to force a zero
1483            ;*****             into sign of B Reg during shift.
1484            ;*****
1485            ;*****     !! Shift opcode is same as value loaded into counter !!
1486            ;*****
1487                      PUSH  H#00E      & AM2901 ,B,SRAMQR,AND,ZB
1488            /                          & CARRYL          ;
1489   000E8    /                          & SHIFT B#1110    ;  B:=0; Q0BUF:=Q0; COUNTER
1490                      RFCT             & AM2901 MPY,B,SRAMQR,ADD,AB
1491            /                          & CARRYL          ;
1492   000E9    /                          & SHIFT B#1110    ;  MULTIPLY STEP
1493                      CONT             & AM2901 MPY,B,SRAMQR,SUBR,AB
1494            /                          & CARRYL          ;
1495            /                          & SHIFT B#1110    ;
1496   000EA    /                          & IFETCH          ;
1497                      JZ               & AM2901 ,A,RAMF,PASS,ZQ
1498            /                          & LODMSR RESET    ; A = Q
1499   000EB    /                          & ENBLO           ;
```

```
LINE    ADDR    STATEMENT

1501            ;*********************************************************************
1502            ;*                                                                   *
1503            ;*        ODDS AND ENDS                                              *
1504            ;*                                                                   *
1505            ;*********************************************************************
1506            ;
1507            INCQ:     RET             & AM2901 ,,QREG,ADD,ZQ
1508            /                         & CARRYH          ; INC Q AND LOAD MAR
1509    000EC   /                         & LDMAR           ; WITH NEXT SEQUENTIAL ADDR
1510            ;
1511            FIXPC:    RET             & AM2901 PC,PC,RAMA,ADD,ZB
1512            /                         & CARRYH          ; INC PC AND LOAD MAR
1513    000ED   /                         & LDMAR           ; WITH OLD PC FOR FETCH
1514            ;
1515            PAT02C:   CONT            & AM2901 PC,PC,NOP,SUBR,DA
1516            /                         & CARRYL          ;
1517            /                         & IMM H#0002      ;
1518    000EE   /                         & LDMAR           ; LOAD MAR WITH RFETCH ADDR
1519                      RET             & AM2901 ,PC,NOP,AND,DQ
1520            /                         & LUSRCOND        ;
1521    000EF   /                         & IMM H#0080      ; TEST GR BIT
```

```
LINE    ADDR    STATEMENT

1523            ;*************************************************************************
1524            ;*                                                                       *
1525            ;*        Interrupt Vector Table                                         *
1526            ;*                                                                       *
1527            ;*************************************************************************
1528            ;
1529    000F0           ALIGN H#10
1530    000F0   ;
1531            INTTBL: EQU     $
1532            ;
1533                    JP      INTPON    & AM2901 B,B,RAMF,EXNOR,AB
1534            /                         & LODMSR RESET    ; 0 - POWER ON
1535    000F0   /                         & ENBLC           ;
1536                    JP      INTFTCH   & AM2901 ,PC,NOP,PASS,ZB
1537            /                         & LDMAR           ;
1538    000F1   /                         & DREAD           ; SET SRCABREF FOR TAB
1539                    JP      INTPARTY  & AM2901 ,,NOP,ADD,AQ
1540    000F2   /                         & ENCN CLRPEI     ; 2 - PARITY ERROR
1541                    JP      INTPROT   & AM2901 ,,NOP,ADD,AQ
1542    000F3   /                         & ENCN CLRMPI     ; 3 - MEMORY PROTECT
1543                    JP      INTSLRQ   & AM2901 ,,NOP,ADD,AQ
1544    000F4   /                         & SPETC SLACK     ; 4 - SLAVE REQUEST
1545                    JP      INTPFW    & AM2901 ,,NOP,ADD,AQ
1546    000F5   /                         & ENCN CLRPFWI    ; 5 - POWER FAIL
1547                    JP      INTTBG    & AM2901 ,,NOP,ADD,AQ
1548    000F6   /                         & ENCN CLRTBT     ; 6 - TBG
1549                    JP      INTIO     & AM2901 ,,NOP,ADD,AQ
1550    000F7   /                         & SPNOP           ; 7 - I/O INTERRUPT
```

```
LINE    ADDR    STATEMENT

1552    000FF           ORG H#0FF
1553    000FF   ;*************************************************************************
1554    000FF   ;*                                                                       *
1555    000FF   ;*       Unimplemented Instruction Entry                                 *
1556    000FF   ;*                                                                       *
1557    000FF   ;*   Note:  Placing entry point at H#FF allows unimplemented            *
1558    000FF   ;*          instructions to map here since blank PROM (7649)             *
1559    000FF   ;*          is all ones.                                                 *
1560    000FF   ;*                                                                       *
1561    000FF   ;*************************************************************************
1562    000FF   ;*****
1563    000FF   ;*****
1564    000FF   ;*****    - UIT
1565    000FF   ;*****    - Algorithm is :  Turn off MGOKILLER
1566    000FF   ;*****                      Set up maps for interrupt handling
1567    000FF   ;*****                      Load CIR and MAR with 8 (trap cell),
1568    000FF   ;*****                        FETCH
1569    000FF   ;*****                      Set TDI so target of trap cell gets
1570    000FF   ;*****                        fetched (done in SETMAPS)
1571    000FF   ;*****                      Turn MGOKILR back on
1572    000FF   ;*****                      JZ to decode trap cell instruction
1573    000FF   ;*****
1574            UIT:      CALL  SETMAPS    & AM2901 ,,NOP,ADD,AQ
1575    000FF   /                          & SPETC MKLROFF  ;
1576                      CONT             & AM2901 ,CIR,RAMF,PASS,DZ
1577            /                          & IMM H#0008     ; TRAP CELL 8
1578            /                          & LDMAR          ; LOAD CIR
1579    00100   /                          & IFETCH         ; FETCH FROM TRAP CELL
1580                      JZ               & AM2901 ,PC,RAMF,SUBR,ZB
1581            /                          & CARRYH         ;
1582    00101   /                          & SPETC MKLRON   ;  PC POINTS TO UIT + 1
```

```
LINE    ADDR    STATEMENT

1584            ;*********************************************************************
1585            ;*                                                                   *
1586            ;*        Interrupt Handler                                          *
1587            ;*                                                                   *
1588            ;*  - Since PC points to next instruction + 1, back up PC            *
1589            ;*                                                                   *
1590            ;*********************************************************************
1591            ;*****
1592            ;*****    - All interrupts come here, except for UIT, which maps
1593            ;*****       directly to EAGUIT at H#0FF
1594            ;*****    - Algorithm is :  Subtract 2 from PC
1595            ;*****                      Turn off MGOKILLER
1596            ;*****                      Vector to the appropriate interrupt
1597            ;*****                        routine
1598            ;*****
1599            INTERRPT: CONT             & AM2901 ,S0,RAMF,EXNOR,DZ
1600   00102    /                          & IMM H#0001      ; S0 = -2
1601                      VECT  INTTBL     & AM2901 S0,PC,RAMF,ADD,AB
1602            /                          & CARRYL          ;
1603   00103    /                          & SPETC MKLROFF   ; PC = PC - 2
```

```
LINE    ADDR    STATEMENT

1605            ;*********************************************************************
1606            ;*                                                                   *
1607            ;*   MPLOWSC - If the select code was one, jump to routine           *
1608            ;*             and perform select code 1 operation,                  *
1609            ;*             else generate a memory protect violation.             *
1610            ;*                                                                   *
1611            ;*********************************************************************
1612            ;*****
1613            MPLOWSC:  CONT            & AM2901 MP,MP,RAMF,AND,DA
1614            /                         & LUSRCOND        ;
1615   00104    /                         & IMM H#003F      ; MASK OFF SELECT CODE
1616                      JRP   GENMPV    & AM2901 PC,PC,NOP,ADD,AQ
1617   00105    /                         & CONDUSR NZ      ; IF SC WAS 1, DO IT; ELSE
1618            ;
1619            ;*********************************************************************
1620            ;*                                                                   *
1621            ;*   GENMPV - Generate memory protect violation                      *
1622            ;*                                                                   *
1623            ;*        - This routine will generate a protect violation           *
1624            ;*          for a priviledged instruction that was executed          *
1625            ;*          with the memory protect system enabled.                  *
1626            ;*        - The procedure is to generate a pending memory            *
1627            ;*          protect interrupt, and restart the instruction           *
1628            ;*          mapping process.                                         *
1629            ;*        - No new fetch is performed.  The idea is to leave         *
1630            ;*          the PC pointing to the word after the offending          *
1631            ;*          instruction.                                             *
1632            ;*                                                                   *
1633            ;*********************************************************************
1634            GENMPV:   JZ              & AM2901 PC,PC,SRAMR,ADD,AB
1635            /                         & CARRYL          ; GENERATE MEMORY PROTECT I
1636            /                         & SHIFT ROTATE    ; SETMPI CLEARS TDI
1637   00106    /                         & ENCN SETMPI     ; CLEAR SIGN BIT OF PC
```

```
LINE   ADDR   STATEMENT

1639          ;*****
1640          ;*****   DIVCONT - CONTINUATION OF DIVIDE ROUTINE
1641          ;*****
1642          ;*****   - Test for negative dividend
1643          ;*****   - Put lower word of dividend in QREG
1644          ;*****
1645          DIVCONT: CJP   DIV05      & AM2901 ,A,QREG,PASS,ZB
1646  00107   /                         & CONDUSR NSGN    ;  IF POSITIVE DIVIDEND
1647          ;*****
1648          ;*****   - Two's complement dividend
1649          ;*****
1650                   CONT             & AM2901 ,,QREG,SUBS,ZQ
1651          /                         & CARRYH          ;
1652  00108   /                         & LODUSR          ;
1653                   CONT             & AM2901 ,B,RAMF,SUBS,ZB
1654  00109   /                         & CARRYUC         ;
1655          ;*****
1656          ;*****   - Put divisor in S2
1657          ;*****   - Set sign and zero in macro status register accordingly
1658          ;*****   - clear macro overflow
1659          ;*****
1660          DIV05:   CONT             & AM2901 TAB,S2,RAMF,ADD,DZ
1661          /                         & CARRYL          ;
1662          /                         & LODMSR          ;
1663  0010A   /                         & ENBLO           ;
1664          ;*****
1665          ;*****   - Test for zero divisor
1666          ;*****   - put most significant word of dividend in S0
1667          ;*****
1668                   CJP   DIVOVFL    & AM2901 B,S0,RAMF,PASS,ZA
1669  0010B   /                         & CONDMSR Z       ; IF DIVISOR=0, OVERFLOW
1670          ;*****
1671          ;*****   - Test for divisor too small (quotient >= 2**16)
1672          ;*****
1673                   CONT             & AM2901 ,S0,NOP,PASS,ZB
1674  0010C   /                         & DIVCOND         ; SET SIGN DIFF FLIP-FLOP
1675                   CJP   DIVOVFL    & AM2901 S2,DIV,RAMF,SUBR,AB
1676          /                         & CARRYL          ;
1677          /                         & CONDUSR NSGN    ;
1678  0010D   /                         & SEQFRZ          ; IF POSITIVE THEN OVERFLOW
1679          ;*****
1680          ;*****   - DIVIDE ITERATION STEP
1681          ;*****
1682          ;*****  Note:  Shift opcode is same as value loaded into 2910
1683          ;*****         counter.  This shift produces a bit in the
1684          ;*****         remainder (B reg) that is discarded (see right
1685          ;*****         shift remainder below), so the bit shifted into
1686          ;*****         the Q reg in the next line is a don't care.
1687          ;*****
1688                   PUSH  H#00F      & AM2901 ,S0,SRAMQL,PASS,ZB
1689          /                         & DIVCOND         ;
1690  0010E   /                         & SHIFT B#1111    ;  SET SIGN DIFF FF
1691                   RFCT             & AM2901 S2,DIV,SRAMQL,SUBR,AB
```

```
LINE    ADDR    STATEMENT

1692            /                       & CARRYL        ;
1693            /                       & DIVCOND       ;
1694    0010F   /                       & SHIFT B#1111  ;
1695            ;*****
1696            ;*****  - Shift remainder right one
1697            ;*****
1698                    CONT            & AM2901 ,S0,SRAMR,PASS,ZB
1699    00110   /                       & SHIFT B#1111  ;  SHIFT IN SIGN
1700            ;*****
1701            ;*****  - Test for remainder negative
1702            ;*****
1703                    CJP   DIV20     & AM2901 ,S0,NOP,PASS,ZB
1704            /                       & CONDUSR NSGN  ;
1705    00111   /                       & SEQFRZ        ; IF POSITIVE THEN OKAY
1706            ;*****
1707            ;*****  - If remainder is negative, restore remainder by
1708            ;*****    adding/subtracting divisor.
1709            ;*****
1710                    CONT            & AM2901 ,S0,NOP,PASS,ZB
1711    00112   /                       & DIVCOND       ; SET SIGN DIFF FF
1712                    CONT            & AM2901 S2,DIV,RAMF,SUBR,AB
1713    00113   /                       & CARRYL        ; RESTORE REMAINDER SIGN
1714            ;*****
1715            ;*****  - If dividend was negative, complement remainder
1716            ;*****  - Compute expected sign : dividend sign XOR divisor sign
1717            ;*****
1718            DIV20:  CONT            & AM2901 ,S1,NOP,PASS,ZB
1719    00114   /                       & LODUSR        ;
1720                    CJP   DIV25     & AM2901 S1,S2,RAMF,EXOR,AB
1721    00115   /                       & CONDUSR NSGN  ;   IF POSITIVE DIVIDEND
1722                    CONT            & AM2901 ,S0,RAMF,SUBS,ZB
1723    00116   /                       & CARRYH        ; COMPLEMENT REMAINDER
1724            ;*****
1725            ;*****  - If dividend and divisor had different sign,
1726            ;*****    complement quotient
1727            ;*****
1728                    CONT            & AM2901 ,S2,NOP,PASS,ZB
1729    00117   /                       & LODUSR        ; TEST EXPECTED SIGN
1730            DIV25:  CJP   DIV30     & AM2901 S0,B,RAMF,PASS,ZA
1731    00118   /                       & CONDUSR SGN   ;  IF DIFFERENT SIGN
1732            ;*****
1733            ;*****  - One's complement quotient to form true quotient
1734            ;*****
1735                    JP    DIV40     & AM2901 PC,A,RAMA,SUBS,ZQ
1736            /                       & CARRYL        ;
1737            /                       & LODUSR        ;
1738    00119   /                       & LDMAR         ; MAR = NEXT INST
1739            ;*****
1740            ;*****  - Change one's complement to two's complement quotient
1741            ;*****
1742            DIV30:  CONT            & AM2901 PC,A,RAMA,ADD,ZQ
1743            /                       & CARRYH        ;
1744            /                       & LODUSR        ;
1745    0011A   /                       & LDMAR         ; MAR = NEXT INST
```

```
LINE    ADDR    STATEMENT

1746            ;*****
1747            ;*****    - Done if quotient is zero
1748            ;*****
1749            DIV40:    CJP    INCPC      & AM2901 A,S2,NOP,EXOR,AB
1750            /                           & CONDUSR Z       ;
1751    0011B   /                           & CMGO & IFETCH   ; SET USR WITH SIGN
1752            ;*****
1753            ;*****    - Calculate overflow
1754            ;*****    - Algorithm is :  Set 0 if  divisor sign XOR quotient
1755            ;*****                                sign XOR dividend sign = 0
1756            ;*****
1757                      CJP    INCPC      & AM2901 ,A,QREG,PASS,ZB
1758            /                           & CONDUSR NSGN    ;
1759    0011C   /                           & CMGO & IFETCH   ;
1760            ;*****
1761            ;*****    - Divide overflow
1762            ;*****    - Set Overflow and put QREG back in A
1763            ;*****
1764            DIVOVFL:  JP     INCPC      & AM2901 PC,A,RAMA,PASS,ZQ
1765            /                           & LODMSR SET      ;
1766            /                           & ENBLO           ;
1767            /                           & LDMAR           ;
1768    0011D   /                           & IFETCH          ; SET OVERFLOW
```

```
LINE    ADDR    STATEMENT

1770            ;*********************************************************************
1771            ;*                                                                   *
1772            ;*        INDIRECT RESOLVERS                                         *
1773            ;*        ------------------                                         *
1774            ;*                                                                   *
1775            ;*    Assumes: - Address word read in progress                       *
1776            ;*                                                                   *
1777            ;*   * Allows base relative and A/B addressability                   *
1778            ;*   * All reads are from data map                                   *
1779            ;*   * Allows three levels of indirection before                     *
1780            ;*       becoming interruptible                                      *
1781            ;*   * The following is the maximum non-interruptible                *
1782            ;*     chain of indirects                                            *
1783            ;*                                                                   *
1784            ;*     +-+---+---+---+---+---+                                       *
1785            ;*     |1|MRG|   |   |   |   |    MRG indirect                       *
1786            ;*     +-+---+---+---+---+---+    (or indirect DEF)                 *
1787            ;*                  |                                                *
1788            ;*                  v                                                *
1789            ;*                                                                   *
1790            ;*     +-+---+---+---+---+---+                                       *
1791            ;*     |1|   |   |   |   |   |    second indirect                    *
1792            ;*     +-+---+---+---+---+---+                                       *
1793            ;*                  |                                                *
1794            ;*                  v                                                *
1795            ;*     +-+---+---+---+---+---+                                       *
1796            ;*     |1|   |   |   |   |   |    third  indirect                    *
1797            ;*     +-+---+---+---+---+---+                                       *
1798            ;*                  |                                                *
1799            ;*                  v                                                *
1800            ;*     +-+---+---+---+---+---+                                       *
1801            ;*     |0|   |   |   |   |   |    actual address                     *
1802            ;*     +-+---+---+---+---+---+                                       *
1803            ;*                  |                                                *
1804            ;*                  v                                                *
1805            ;*     +-+---+---+---+---+---+                                       *
1806            ;*     | |   |   |   |   |   |    data                               *
1807            ;*     +-+---+---+---+---+---+                                       *
1808            ;*                                                                   *
1809            ;*                                                                   *
1810            ;*********************************************************************
```

```
LINE    ADDR    STATEMENT

1812            ;*********************************************************************
1813            ;*                                                                   *
1814            ;*       Resolve indirect with operand read;  The resolved           *
1815            ;*       address is left in the MAR and also in QREG.                *
1816            ;*                                                                   *
1817            ;*       The return is executed at the same time that the            *
1818            ;*       operand read is started.  This means that the first         *
1819            ;*       cycle after return is free for another operation            *
1820            ;*                                                                   *
1821            ;*       Note: This means that if the data is wanted, you must       *
1822            ;*       either wait a cycle or code a MWAIT or a MGO special.       *
1823            ;*                                                                   *
1824            ;*       This is the pure indirect resolver, with read of            *
1825            ;*       operand.                                                    *
1826            ;*                                                                   *
1827            ;*********************************************************************
1828            ;*****
1829            ;*****    - Load MAR with next address
1830            ;*****    - Start read of operand
1831            ;*****    - Return if this address being loaded is direct
1832            ;*****
1833            READRES:  CRET            & AM2901 BR,TABQ,QREG,ADD,DZ
1834            /                         & CARRYL         ;
1835            /                         & CONDEXT NSIGN  ;
1836            /                         & LDMAR          ;
1837   0011E    /                         & DREAD          ;
1838            ;*****
1839            ;*****    - Wait for previous read to finish
1840            ;*****    - Load MAR with next address
1841            ;*****    - Start read of operand
1842            ;*****    - Return if the address being loaded is direct
1843            ;*****
1844                      CRET            & AM2901 BR,TABQ,QREG,ADD,DZ
1845            /                         & CARRYL         ;
1846            /                         & CONDEXT NSIGN  ;
1847            /                         & LDMAR          ;
1848   0011F    /                         & DREAD          ;
1849                      LDCT  RDRES2    & AM2901 ,,NOP,ADD,AQ
1850   00120    /                         & SPNOP          ;
1851            ;*****
1852            ;*****    - Load MAR with next address
1853            ;*****    - Start read of operand
1854            ;*****    - Return if address being loaded is direct
1855            ;*****
1856            RDRES2:   CRET            & AM2901 BR,TABQ,QREG,ADD,DZ
1857            /                         & CARRYL         ;
1858            /                         & CONDEXT NSIGN  ;
1859            /                         & LDMAR          ;
1860   00121    /                         & DREAD          ;
1861                      JRP   INTERRPT  & AM2901 ,,NOP,ADD,AQ
1862   00122    /                         & CONDEXT INTRPT ; IF INT PEND, JP
```

```
LINE    ADDR    STATEMENT

1864            ;*********************************************************************
1865            ;*                                                                   *
1866            ;*        READIND (MRGIND)                                           *
1867            ;*                                                                   *
1868            ;*        Resolve indirect with operand read.  Upon exit, the        *
1869            ;*        MAR will contain PC-1 if an MRG instruction is being       *
1870            ;*        executed, or PC if a non-MRG instruction is being          *
1871            ;*        executed.  QREG is loaded with the resolved address.       *
1872            ;*                                                                   *
1873            ;*        READIND is also useful for resolving the second word       *
1874            ;*        of a multi-word instruction, since it leaves PC            *
1875            ;*        (=next_inst + 1) in the MAR upon exit                      *
1876            ;*********************************************************************
1877            ;*****
1878            ;*****    - Put PC or PC-1 in MAR
1879            ;*****    - Return if last address loaded was direct
1880            ;*****
1881            READIND:  CRET              & AM2901 ,PC,NOP,SUBR,ZB
1882            /                           & CARRYL         ;
1883            /                           & CONDUSR NSGN   ;
1884    00123   /                           & LDMAR          ; SUBTRACTS 1 IF MRG
1885            ;*****
1886            ;*****    - Load 2910 counter with 2 (loop 3 times)
1887            ;*****    - Load MAR with next address
1888            ;*****    - Read next word
1889            ;*****
1890            ;*****          *** NOTE ***
1891            ;*****      MRGIND MUST ALIGN SO THAT LOW 4 BITS ARE B#0100
1892            ;*****
1893            MRGIND:   LDCT  H#002       & AM2901 BR,TABQ,QREG,ADD,DZ
1894            /                           & CARRYL         ;
1895            /                           & LODUSR         ;
1896            /                           & LDMAR          ;
1897    00124   /                           & DREAD          ; READ NEXT INDIRECT OR OPE
1898            MRGIND2:  CRET              & AM2901 ,PC,NOP,SUBR,ZB
1899            /                           & CARRYL         ;
1900            /                           & CONDUSR NSGN   ;
1901    00125   /                           & LDMAR          ; SUBTRACTS 1 IF MRG
1902                      RPCT  MRGIND2     & AM2901 BR,TABQ,QREG,ADD,DZ
1903            /                           & CARRYL         ;
1904            /                           & LODUSR         ;
1905            /                           & LDMAR          ;
1906    00126   /                           & DREAD          ;
1907                      CJPP  INTERRPT    & AM2901 ,,NOP,ADD,AQ
1908    00127   /                           & CONDEXT INTRPT ;
1909            ;*****
1910            ;*****    - Jump to READIND to do 3 more levels
1911            ;*****    - Must pass status of address (=QREG) to USR for
1912            ;*****       READIND test
1913            ;*****
1914                      JP    READIND     & AM2901 ,,NOP,PASS,ZQ
1915    00128   /                           & LODUSR         ; MUST PASS QREG TO USR
```

```
LINE    ADDR    STATEMENT

1917            ;*********************************************************************
1918            ;*                                                                   *
1919            ;*       Resolve indirect without operand read, and leave            *
1920            ;*       resolved address in MAR.  The resolved address is           *
1921            ;*       also left in QREG.                                          *
1922            ;*                                                                   *
1923            ;*       This indirect resolver is used if the address is a          *
1924            ;*       store address and the target word should not be read.       *
1925            ;*                                                                   *
1926            ;*********************************************************************
1927            ;*****
1928            ;*****    - Load QREG and MAR with next address
1929            ;*****    - If address is indirect, do a DREAD
1930            ;*****    - if address is direct, return
1931            ;*****
1932            INDRES:   CRET            & AM2901 BR,TABQ,QREG,ADD,DZ
1933            /                         & CARRYL          ;
1934            /                         & CONDEXT         ;
1935            /                         & LDMAR           ;
1936            /                         & MWAIT           ;  MWAIT == CONDEXT NSIGN
1937   00129    /                         & CMSGN & DREAD   ;  READS IF INDIRECT; RTN I
1938            ;*****
1939            ;*****    - If address is indirect, do a DREAD
1940            ;*****    - If address is direct, return
1941            ;*****
1942            INDRES1:  CRET            & AM2901 BR,TABQ,QREG,ADD,DZ
1943            /                         & CARRYL          ; RETURN IF NOT INDIRECT
1944            /                         & CONDEXT         ; CONDEXT IS NSIGN (=MWAIT)
1945            /                         & LDMAR           ; MAR CONTAINS ADDRESS ON R
1946            /                         & MWAIT           ; WAIT FOR READ
1947   0012A    /                         & CMSGN & DREAD   ; READS IF INDIRECT
1948                      LDCT  INDRES2   & AM2901 ,,NOP,ADD,AQ
1949   0012B    /                         & SPNOP           ; LOAD COUNTER FOR NO INTRP
1950            ;*****
1951            ;*****    - Load QREG and MAR with next address
1952            ;*****    - If address is indirect, do a DREAD
1953            ;*****    - If address is direct, return
1954            ;*****
1955            INDRES2:  CRET            & AM2901 BR,TABQ,QREG,ADD,DZ
1956            /                         & CARRYL          ;
1957            /                         & CONDEXT         ;
1958            /                         & LDMAR           ;
1959            /                         & MWAIT           ; MWAIT == CONDEXT NSIGN
1960   0012C    /                         & CMSGN & DREAD   ; READS IF INDIRECT; RTN IF
1961            ;*****
1962            ;*****    - Jump to INDRES2, or to INTERRPT if there is an
1963            ;*****        interrupt pending
1964            ;*****
1965                      JRP   INTERRPT  & AM2901 ,,NOP,ADD,AQ
1966   0012D    /                         & CONDEXT INTRPT  ;
```

```
LINE    ADDR    STATEMENT

1968            ;
1969            ASLOVFL:  RET             & AM2901 ,S0,SRAML,PASS,DZ
1970    0012E   /                         & IMM H#8000      ; SHIFT B#0000 IMPLIED
1971            ;                                           ; SETS MC (=M0 AFTER SWAPE0
1972            ;                      *** NOTE ***
1973            ;         JZA MUST ALIGN WITH LOW 4 BITS = B#1111
1974            ;
1975            JZA:      JZ              & AM2901 ,A,RAMF,PASS,ZQ
1976    0012F   /                         & SPNOP           ; A = QREG
1977    00130             FILLER
1978    00131             FILLER
1979    00132             FILLER
1980    00133             FILLER
1981    00134             FILLER
```



```
LINE    ADDR    STATEMENT

1983            ;*************************************************************************
1984            ;*                                                                       *
1985            ;*        I/O Instruction Subroutines                                    *
1986            ;*                                                                       *
1987            ;*************************************************************************
1988            ;*****
1989            ;*****    - IOHSHAKE - I/O chip handshake
1990            ;*****
1991            ;*****      This routine handshakes a word from an I/O chip.
1992            ;*****      It will wait 8 microcycles before aborting the
1993            ;*****      handshake and fetching the next instruction.
1994            ;*****
1995            IOHSHAKE: PUSH  H#002     & AM2901 ,,NOP,ADD,AQ
1996    00135   /                         & SPNOP           ; WAIT FOR I/O
1997                      RFCT            & AM2901 ,,NOP,ADD,AQ
1998    00136   /                         & SPNOP           ; WAIT FOR I/O
1999            IOHSHAK0: PUSH  H#002     & AM2901 ,,NOP,ADD,AQ
2000    00137   /                         & SPNOP           ;
2001                      TWB   FIXMAR    & AM2901 ,,NOP,ADD,AQ
2002    00138   /                         & CONDEXT IORQ    ; WAIT FOR IORQ
2003            ;*****
2004            ;*****    - I/O chip asserted IORQ
2005            ;*****    - Read the control word
2006            ;*****
2007            IOHSHAK1: CONT            & AM2901 ,,NOP,ADD,AQ
2008    00139   /                         & SPETC IORD      ;
2009            IOHSHAK2: CJP   IOHSHAK2  & AM2901 ,,NOP,ADD,AQ
2010    0013A   /                         & CONDEXT IORQ    ; WAIT FOR IORQ TO GO AWAY
2011            ;*****
2012            ;*****    - Control word is now in the MDIR and in the IR
2013            ;*****    - Put control word in QREG
2014            ;*****    - Return
2015            ;*****
2016                      RET             & AM2901 PC,,QREG,PASS,DZ
2017    0013B   /                         & SPNOP           ;
```

```
LINE    ADDR    STATEMENT

2019            ;*********************************************************************
2020            ;*                                                                   *
2021            ;*        I/O CONTROL WORD ROUTINES                                  *
2022            ;*                                                                   *
2023            ;*        The control word is passed over the data bus on            *
2024            ;*        bits DB4-DB7 with DB8 being the continue bit               *
2025            ;*                                                                   *
2026            ;*        The control words are :                                    *
2027            ;*                                                                   *
2028            ;*      * 0000 - NOP                                                 *
2029            ;*      * 0001 - LOAD P FROM DATA BUS                                *
2030            ;*      * 0010 - LOAD A FROM DATA BUS                                *
2031            ;*      * 0011 - LOAD B FROM DATA BUS                                *
2032            ;*        0100 - CLEAR THE OVERFLOW BIT                              *
2033            ;*        0101 - SET THE OVERFLOW BIT                                *
2034            ;*      * 0110 - MERGE INTO A REG FROM DATA BUS                      *
2035            ;*      * 0111 - INCREMENT P                                         *
2036            ;*        1000 - PUT STATUS REG ON DATA BUS                          *
2037            ;*      * 1001 - ENABLE BOOT ROM                                     *
2038            ;*      * 1010 - PUT A ON DATA BUS                                   *
2039            ;*      * 1011 - PUT B ON DATA BUS                                   *
2040            ;*        1100 - CLEAR E REG                                         *
2041            ;*        1101 - SET E REG                                           *
2042            ;*      * 1110 - PUT P ON DATA BUS                                   *
2043            ;*        1111 - PUT P ON DATA BUS, INCREMENT P                      *
2044            ;*                                                                   *
2045            ;*        NOTE - ONLY THE STARRED (*) CONTROL WORDS ARE              *
2046            ;*        GENERATED BY THE I/O MASTER, AND SO THEY ARE THE           *
2047            ;*        ONLY ONES IMPLEMENTED                                      *
2048            ;*                                                                   *
2049            ;*********************************************************************
2050            ;*****
2051            ;*****    - Undefined control word is a NOP
2052            ;*****
2053            IODCXXXX: RET             & AM2901 PC,,NOP,ADD,AQ
2054   0013C    /                         & SPNOP          ; IF UNDEFINED OR NOT IMPLE
2055            ;*****
2056            ;*****    - Load P from data bus
2057            ;*****
2058            IODC0001: CALL  IOHSHAK0  & AM2901 ,,NOP,ADD,AQ
2059   0013D    /                         & SPNOP          ; LOAD P FROM DATA BUS
2060                      RET             & AM2901 PC,PC,RAMF,PASS,DZ
2061   0013E    /                         & SPNOP          ; RETURN
2062            ;*****
2063            ;*****    - Load A from data bus
2064            ;*****
2065            IODC0010: CALL  IOHSHAK0  & AM2901 ,,NOP,ADD,AQ
2066   0013F    /                         & SPNOP          ; LOAD A FROM DATA BUS
2067                      RET             & AM2901 PC,A,RAMF,PASS,DZ
2068   00140    /                         & SPNOP          ;
2069            ;*****
2070            ;*****    - Load B from data bus
2071            ;*****
```

```
LINE    ADDR    STATEMENT

2072            IODC0011: CALL  IOHSHAK0   & AM2901 ,,NOP,ADD,AQ
2073    00141   /                          & SPNOP          ; LOAD B FROM DATA BUS
2074                      RET              & AM2901 PC,B,RAMF,PASS,DZ
2075    00142   /                          & SPNOP          ;
2076            ;*****
2077            ;*****    - Merge into A/B
2078            ;*****    - Note : A/B is selected by IR bit 10
2079            ;*****
2080            IODC0110: CALL  IOHSHAK0   & AM2901 ,,NOP,ADD,AQ
2081    00143   /                          & SPNOP          ; MERGE INTO A/B (IR10)
2082                      RET              & AM2901 CAB,CAB,RAMF,OR,DA
2083    00144   /                          & SPNOP          ;
2084            ;*****
2085            ;*****    - Increment PC
2086            ;*****
2087            IODC0111: RET              & AM2901 PC,PC,RAMF,ADD,ZB
2088            /                          & CARRYH         ;
2089    00145   /                          & SPNOP          ; INCREMENT PC
2090            ;*****
2091            ;*****    - Enable VCP
2092            ;*****
2093            ;*****    - In order to enable the VCP, the following operations
2094            ;*****        must be done:
2095            ;*****
2096            ;*****          - Format and save interrupted Q in IQLOC
2097            ;*****          - Format and save IMAP in IMAPLOC
2098            ;*****          - Turn off memory protect system
2099            ;*****          - Enable boot memory
2100            ;*****
2101            IODC1001: CONT             & AM2901 ,S0,SRAMR,PASS,DZ
2102            /                          & IMM H#0040     ; S0 = H#0020
2103    00146   /                          & LDMAR          ; SAVE WMAP IN BOOT MEM LOC
2104                      CALL  STWMAP     & AM2901 ,S0,NOP,PASS,ZB
2105    00147   /                          & LDAER          ; AER := H#20 (BOOT MEM)
2106                      RET              & AM2901 PC,MAPX,NOP,PASS,DZ
2107            /                          & IMMB H#60      ; MAPX = BOOT MEMORY, ABREF
2108    00148   /                          & ENCN CLRMPEN   ; TURN OFF MP SYSTEM
2109            ;*****
2110            ;*****    - Put A register on the data bus
2111            ;*****    - Note : The return from IOHSHAK2 returns directly to
2112            ;*****        the calling routine
2113            ;*****
2114            IODC1010: JP    IOHSHAK2   & AM2901 A,A,RAMA,PASS,ZB
2115            /                          & LDMDOR         ;
2116    00149   /                          & SPETC IOWR     ; PLACE A ON DATA BUS
2117            ;*****
2118            ;*****    - Put B register on the data bus
2119            ;*****    - Note : The return from IOHSHAK2 returns directly to
2120            ;*****      the calling routine
2121            ;*****
2122            IODC1011: JP    IOHSHAK2   & AM2901 B,B,RAMA,PASS,ZB
2123            /                          & LDMDOR         ;
2124    0014A   /                          & SPETC IOWR     ; PLACE B ON DATA BUS
2125            ;*****
```

```
LINE    ADDR    STATEMENT

2126            ;*****    - Put PC on the data bus
2127            ;*****    - Note : The return from IOHSHAK2 returns directly to
2128            ;*****      the calling routine
2129            ;*****
2130            IODC1110: JP    IOHSHAK2   & AM2901 PC,PC,RAMA,PASS,ZB
2131            /                          & LDMDOR          ;
2132    0014B   /                          & SPETC IOWR      ; PLACE PC ON DATA BUS
```

```
LINE    ADDR    STATEMENT

2134            ;*********************************************************************
2135            ;*                                                                   *
2136            ;*        - Power on interrupt                                       *
2137            ;*        - Do a basic self-test of machine operation                *
2138            ;*        - A pattern of all ones with a single zero bit is          *
2139            ;*          rotated thru all data bus bits.                          *
2140            ;*                                                                   *
2141            ;*********************************************************************
2142            INTPON:   PUSH            & AM2901 B,A,SRAML,PASS,ZA
2143            /                         & ENCN SETDTST   ;
2144   0014C    /                         & SHIFT ROTATEC  ; A := H#FFFE
2145                      CONT            & AM2901 ,A,SRAML,PASS,ZB
2146            /                         & SHIFT ROTATE   ;
2147            /                         & LODUSR         ;
2148   0014D    /                         & LDMDOR         ; WRITE PATTERN IN A
2149                      LOOP            & AM2901 B,B,RAMA,EXNOR,DA
2150            /                         & CONDUSR NSGN   ;
2151   0014E    /                         & LDMAR          ; BUILD CHECK PATTERN IN B
2152                      CJP   LASTWD    & AM2901 ,A,RAMF,AND,ZB
2153            /                         & CARRYH         ;
2154   0014F    /                         & CONDUSR Z      ; IF B=0 THEN PASS
2155            ;*****
2156            ;*****    - Self-test failure
2157            ;*****
2158            INTDEAD:  CONT            & AM2901 ,A,SRAML,PASS,ZB
2159            /                         & SHIFT ROTATEC  ;
2160   00150    /                         & LDMDOR         ; GENERATE PATTERN
2161                      JP    INTDEAD   & AM2901 A,A,RAMA,PASS,ZA
2162   00151    /                         & SPNOP          ; LOOP ON PATTERN
2163            ;*****
2164            ;*****    - Self-test passed
2165            ;*****    - Initialize registers to their power-up values
2166            ;*****
2167            ;*****          MAPX  - Points to boot memory
2168            ;*****          MAPD1 - Points to map zero
2169            ;*****          MAPD2 - Points to map zero
2170            ;*****          PC    - Points to location 20002Q in boot memory
2171            ;*****          LEDS  - Result of self-test
2172            ;*****
2173            INTPONOK: CONT            & AM2901 ,MAPD1,NOP,AND,ZQ
2174   00152    /                         & ENCN CLRDTST   ;  TURN OFF DATA BUS LOOPBA
2175            ;                                             AND INITIALIZE MAPD1
2176                      CONT            & AM2901 ;,NOP,PASS,DZ
2177            /                         & LDST           ;
2178   00153    /                         & IMM H#0020     ; INIT PROCESSOR STATUS REG
2179                      CALL  CHECKSUM  & AM2901 ,B,RAMF,PASS,ZB
2180   00154    /                         & SPWR LEDWR     ; SET LEDS
2181                      CONT            & AM2901 ,PC,RAMF,PASS,DZ
2182   00155    /                         & IMM H#2002     ; INIT PC TO 20002Q
2183                      CONT            & AM2901 ,MAPD2,RAMF,AND,ZB
2184   00156    /                         & SPETC MKLRON   ;  S4 := 0; MAPD2 := 0
2185            ;                                             TURN ON MGOKILLER
2186            ;*****
```

```
LINE    ADDR    STATEMENT

2187            ;*****    - Initialize map registers on the memory controller
2188            ;*****    - All map registers are initialized to their own number
2189            ;*****        EX:  MAP0000 := 0;
2190            ;*****             MAP0001 := 1;
2191            ;*****                ...
2192            ;*****             MAP1023 := 1023;
2193            ;*****
2194                      PUSH  H#01F       & AM2901 ,S0,RAMF,AND,ZB
2195    00157   /                           & LODMSR RESET    ;  2910 := NUMBER OF MAP SE
2196            ;*****
2197            ;*****    - Note : The following CPUSH never loads the counter
2198            ;*****
2199                      CPUSH             & AM2901 S0,S0,RAMA,ADD,ZA
2200            /                           & CARRYH          ;
2201            /                           & CONDMSR SGN     ;
2202    00158   /                           & LDAER           ; LOAD AER WITH MAP NUMBER
2203                      CONT              & AM2901 S0,S0,RAMA,ADD,DA
2204            /                           & CARRYL          ;
2205            /                           & LDMAR           ;
2206    00159   /                           & IMM H#0400      ; LOAD MAR WITH MAP REG #
2207                      CONT              & AM2901 S4,S4,RAMA,ADD,ZB
2208            /                           & CARRYH          ;
2209    0015A   /                           & SPWR MAPWR      ; WRITE TO MAP REG
2210                      LOOP              & AM2901 ,S0,NOP,PASS,ZB
2211            /                           & CONDUSR SGN     ;
2212    0015B   /                           & SEQFRZ          ;
2213                      RFCT              & AM2901 S0,S0,RAMF,AND,DA
2214    0015C   /                           & IMM H#001F      ; ZERO MAP REG NUMBER
2215                      CONT              & AM2901 PC,MAPX,NOP,PASS,DZ
2216            /                           & LDAER           ;
2217    0015D   /                           & IMM H#0060      ; SET MAPX TO BOOT MEMORY
2218            ;*****
2219            ;*****    - Fetch the first instruction
2220            ;*****
2221            SKIP:     JZ                & AM2901 PC,PC,RAMA,ADD,ZB
2222            /                           & CARRYH          ;
2223            /                           & LDMAR           ;
2224    0015E   /                           & IFETCH          ; PC -> NEXT_INSTRUCTION
```

```
LINE    ADDR    STATEMENT

2226            ;*********************************************************************
2227            ;*                                                                   *
2228            ;*        - A/B Instruction Fetch Interrupt                          *
2229            ;*        - Interrupt is generated when MAR contains                 *
2230            ;*          zero or one and IFETCH is asserted.  This                *
2231            ;*          condition is latched for the source special.             *
2232            ;*          After interrupt vectoring, the appropriate               *
2233            ;*          register is written to memory and refetched.             *
2234            ;*                                                                   *
2235            ;*********************************************************************
2236            INTFTCH:  CONT              & AM2901 TAB,,NOP,PASS,DZ
2237            /                           & LDMDOR          ;
2238   0015F    /                           & DWRITE          ; WRITE INST TO MEMORY
2239                      CONT              & AM2901 PC,PC,RAMF,ADD,ZB
2240            /                           & CARRYH          ; INC PC
2241   00160    /                           & ABFETCH         ; REFETCH INST FROM MEMORY
2242                      JZ                & AM2901 PC,PC,SRAMR,ADD,AB
2243            /                           & CARRYL          ;
2244            /                           & SHIFT B#0000    ;
2245   00161    /                           & SPETC MKLRON    ;  TURN MGOKILLER BACK ON
2246            ;*****
2247            ;*****    - Parity Error Interrupt
2248            ;*****
2249            INTPARTY: CALL  SETMAPS     & AM2901 ,,NOP,ADD,AQ
2250   00162    /                           & ENCN CLRMPI     ; CLEAR POSS MP INTERRUPT
2251                      CONT              & AM2901 PC,CIR,NOP,PASS,DZ
2252            /                           & IMM H#0005      ;
2253            /                           & LDMAR           ;
2254   00163    /                           & IFETCH          ; FETCH FROM LOCATION 5
2255                      JZ                & AM2901 PC,PC,SRAMR,ADD,AB
2256            /                           & CARRYL          ;
2257            /                           & SHIFT B#0000    ;
2258   00164    /                           & SPETC MKLRON    ;  TURN MKLR BACK ON
2259            ;*****
2260            ;*****    - TBG interrupt
2261            ;*****
2262            INTTBG:   CALL  SETMAPS     & AM2901 ,,NOP,ADD,AQ
2263   00165    /                           & SPNOP           ;
2264                      CONT              & AM2901 PC,CIR,RAMF,PASS,DZ
2265            /                           & IMM H#0006      ;
2266            /                           & LDMAR           ;
2267   00166    /                           & IFETCH          ; FETCH FROM LOC 6
2268                      JZ                & AM2901 PC,PC,SRAMR,ADD,AB
2269            /                           & CARRYL          ;
2270            /                           & SHIFT B#0000    ;
2271   00167    /                           & SPETC MKLRON    ;  TURN MKLR BACK ON
2272            ;*****
2273            ;*****    - I/O interrupt
2274            ;*****
2275            INTIO:    CALL  SETMAPS     & AM2901 ,,NOP,ADD,AQ
2276   00168    /                           & SPETC MIAK      ; ACK I/O INT; SET UP MAPS
2277                      CONT              & AM2901 PC,CIR,NOP,PASS,DZ
2278   00169    /                           & SPRD ECIRRD     ; PUT I/O SELECT CODE IN CI
```

```
LINE    ADDR    STATEMENT

2279                    JZ              & AM2901 PC,PC,SRAMR,ADD,AB
2280            /                       & CARRYL        ;
2281            /                       & SHIFT B#0000  ;
2282    0016A   /                       & SPETC MKLRON  ;  TURN MKLR BACK ON
2283            ;*****
2284            ;*****  - Slave request pseudo-interrupt
2285            ;*****  - Simply handshake I/O control words until the continue
2286            ;*****      bit is no longer set
2287            ;*****
2288            INTSLRQ: CALL  IOHSHAKE & AM2901 ,,NOP,ADD,AQ
2289    0016B   /                       & ENCN SETTDI   ;
2290                    CALL            & AM2901 MP,MP,RAMF,PASS,ZQ
2291            /                       & SPETC IRSP    ; SAVE CONTROL WORD IN MP,
2292    0016C   /                       & JTAB CWDCODE  ; DO THE CONTROL WORD
2293                    CONT            & AM2901 MP,MP,RAMF,AND,DA
2294            /                       & LUSRCOND      ;
2295    0016D   /                       & IMM H#0100    ; TEST LOOP BIT
2296                    CJP   INTSLRQ   & AM2901 PC,PC,NOP,ADD,AQ
2297    0016E   /                       & CONDUSR NZ    ; IF LOOP BIT WAS SET
2298                    JZ              & AM2901 PC,PC,RAMA,ADD,ZB
2299            /                       & CARRYH        ;
2300            /                       & SPETC MKLRON  ;
2301            /                       & LDMAR         ;
2302    0016F   /                       & IFETCH        ; GET NEXT INSTRUCTION
2303            ;*****
2304            ;*****  - Memory protect interrupt
2305            ;*****
2306            INTPROT: CALL  SETMAPS  & AM2901 ,,NOP,ADD,AQ
2307    00170   /                       & SPNOP         ; SAVE MAPS, SET UP NEW MAP
2308                    CONT            & AM2901 PC,CIR,NOP,PASS,DZ
2309            /                       & IMM H#0007    ;
2310            /                       & LDMAR         ;
2311    00171   /                       & IFETCH        ; FETCH FROM LOCATION 7
2312                    JZ              & AM2901 PC,PC,SRAMR,ADD,AB
2313            /                       & CARRYL        ;
2314            /                       & SHIFT B#0000  ;
2315    00172   /                       & SPETC MKLRON  ;  TURN MKLR BACK ON
2316            ;*****
2317            ;*****  - Power fail warning interrupt
2318            ;*****
2319            INTPFW:  CALL  SETMAPS  & AM2901 ,,QREG,PASS,DZ
2320    00173   /                       & SPRD STRD     ; PUT STATUS REG IN QREG
2321                    CONT            & AM2901 ,,NOP,OR,DQ
2322            /                       & IMMB H#08     ;
2323    00174   /                       & LDST          ; INHIBIT TYPE 2 AND 3 INTS
2324                    CONT            & AM2901 PC,CIR,NOP,PASS,DZ
2325            /                       & IMM H#0004    ;
2326            /                       & LDMAR         ;
2327    00175   /                       & IFETCH        ; FETCH FROM LOC 4
2328                    JZ              & AM2901 PC,PC,SRAMR,ADD,AB
2329            /                       & CARRYL        ;
2330            /                       & SHIFT B#0000  ;
2331    00176   /                       & SPETC MKLRON  ;  TURN MKLR BACK ON
```

```
LINE    ADDR    STATEMENT

2333            ;*****************************************************************
2334            ;*                                                               *
2335            ;*        - SETMAPS - set up routine for interrupt handling      *
2336            ;*                                                               *
2337            ;*            - Save current working map set in IMAPLOC          *
2338            ;*            - Save interrupted Q in IQLOC                      *
2339            ;*            - Disable Memory Protect system                    *
2340            ;*            - Turn off CSMODE                                  *
2341            ;*            - Set DATA1 map set to current MAPX                *
2342            ;*            - Set MAPX to 0                                    *
2343            ;*            - Set TDI                                          *
2344            ;*                                                               *
2345            ;*****************************************************************
2346            SETMAPS:  CALL  SAVEINT     & AM2901 ,,NOP,ADD,AQ
2347   00177    /                           & ENCN SETTDI     ; SAVE IMAP, IQ; SET TDI
2348                      CONT              & AM2901 MAPX,MAPD1,RAMF,PASS,DZ
2349   00178    /                           & ENCN CLRMPEN    ;  CLEAR MP SYSTEM FF;
2350            ;                                               S4 = DATA1 = OLD MAPX
2351                      RET               & AM2901 S4,MAPX,NOP,AND,DA
2352            /                           & SPETC CLRCS     ; TURN OFF CSMODE
2353            /                           & IMMB H#60       ;
2354   00179    /                           & LDAER           ; MAPX := 0 (40 IF BOOT MOD
2355            ;*****
2356            ;*****    - SAVEINT - Format and store IMAP and IQ in IMAPLOC and IQLOC
2357            ;*****      respectively.
2358            ;*****
2359            SAVEINT:  CONT              & AM2901 ,,NOP,PASS,DZ
2360            /                           & IMMB H#20       ;
2361   0017A    /                           & LDAER           ; POINT TO BOOT MEMORY
2362                      CONT              & AM2901 Q,,NOP,PASS,DZ
2363            /                           & IMM IQLOC       ;
2364   0017B    /                           & LDMAR           ; ITS GOING IN IQLOC
2365                      CJP   WRITEIQ     & AM2901 Q,S0,RAMF,PASS,ZA
2366   0017C    /                           & CONDEXT CSON    ; IS CS MODE ON?
2367                      CONT              & AM2901 S0,S0,RAMF,OR,DA
2368   0017D    /                           & IMM H#8000      ; NO - SET CSMODE BIT
2369            WRITEIQ:  CONT              & AM2901 S0,TAB,NOP,PASS,ZA
2370            /                           & SPETC CLRCS     ;
2371            /                           & LDMDOR          ;
2372   0017E    /                           & DWRITE          ; TURN CSMODE OFF ; WRITE I
2373            SAVEIMAP: CONT              & AM2901 ,,NOP,PASS,DZ
2374            /                           & IMM IMAPLOC     ;
2375   0017F    /                           & LDMAR           ; NEXT IS IMAP
2376            STWMAP:   CONT              & AM2901 S6,S4,RAMF,PASS,DZ
2377   00180    /                           & IMM H#001F      ; 5 BIT MASK FOR MAP NUMBER
2378                      CONT              & AM2901 MAPD2,S0,SRAML,AND,DA
2379   00181    /                           & SHIFT B#0010    ; S0 := MAPD2*MASK, SL1
2380                      CONT              & AM2901 S0,S0,RAMA,PASS,DZ
2381   00182    /                           & SPRD RL4        ; ROTATE LEFT 4 MORE
2382                      CONT              & AM2901 MAPD1,S5,RAMF,AND,DA
2383   00183    /                           & SPNOP           ; S5 := MAPD1*MASK
2384                      CONT              & AM2901 S5,S0,SRAML,OR,AB
2385   00184    /                           & SHIFT B#0010    ; OR IT IN, SL1
```

```
LINE    ADDR    STATEMENT

2386                    CONT            & AM2901 S0,S0,RAMA,PASS,DZ
2387    00185   /                       & SPRD RL4          ; ROT L 4 MORE
2388                    CONT            & AM2901 MAPX,S5,RAMF,AND,DA
2389    00186   /                       & SPNOP             ; S5 := MAPX*MASK
2390                    CJP  SVWMAP     & AM2901 S5,S0,RAMF,OR,AB
2391    00187   /                       & CONDEXT NMPEN  ; OR IN MAPX; IF MP NOT ON,
2392                    RET             & AM2901 S0,S0,RAMF,OR,DA
2393            /                       & IMM H#8000        ;
2394            /                       & LDMDOR            ;
2395    00188   /                       & DWRITE            ; SET MP BIT; WRITE IMAP; R
2396            SVWMAP: RET             & AM2901 S0,TAB,NOP,PASS,ZA
2397            /                       & LDMDOR            ;
2398    00189   /                       & DWRITE            ; WRITE WMAP; RETURN
```

```
LINE    ADDR    STATEMENT

2400            ;*********************************************************************
2401            ;*                                                                   *
2402            ;*        Index Register Group                                       *
2403            ;*        --------------------                                       *
2404            ;*                                                                   *
2405            ;*********************************************************************
2406            ;*****
2407            ;*****    - ADX. :  Add (MEM) to X/Y
2408            ;*****
2409            ADX.:     CCALL READRES     & AM2901 BR,TABQ,QREG,ADD,DZ
2410            /                           & CARRYL          ;
2411            /                           & CONDEXT MDIR15  ;
2412            /                           & LDMAR           ;
2413   0018A    /                           & DREAD           ; GET OPERAND
2414                      CONT              & AM2901 PC,PC,RAMA,ADD,ZB
2415            /                           & CARRYH          ;
2416   0018B    /                           & LDMAR           ; MAR = NEXT INST
2417                      CONT              & AM2901 TAB,S0,RAMF,PASS,DZ
2418   0018C    /                           & IFETCH          ; OPERAND IN S0 ; FETCH
2419                      JZ                & AM2901 S0,CXY,RAMF,ADD,AB
2420            /                           & CARRYL          ;
2421            /                           & LODMSR ENVE     ;
2422   0018D    /                           & ENBLC & ENBLO   ;
2423            ;*****
2424            ;*****    - Copy A/B to/from X/Y
2425            ;*****
2426            COPYABXY: JZ                & AM2901 CAB,CXY,RAMF,PASS,ZA
2427   0018E    /                           & IFETCH          ;
2428            COPYXYAB: JZ                & AM2901 CXY,CAB,RAMF,PASS,ZA
2429   0018F    /                           & IFETCH          ;
2430            ;*****
2431            ;*****    - Dec/inc X/Y and skip if zero
2432            ;*****
2433            DSXY:     JP    SKIFZ       & AM2901 CXY,CXY,RAMF,SUBR,ZA
2434            /                           & CARRYH          ;
2435   00190    /                           & LODUSR          ; DECREMENT X/Y
2436            ISXY:     JP    SKIFZ       & AM2901 CXY,CXY,RAMF,ADD,ZA
2437            /                           & CARRYH          ;
2438   00191    /                           & LODUSR          ; INCREMENT X/Y
2439            ;*****
2440            ;*****    - JLY - Jump and load Y
2441            ;*****
2442            JLY:      CCALL INDRES      & AM2901 BR,TABQ,QREG,ADD,DZ
2443            /                           & CARRYL          ;
2444            /                           & CONDEXT MDIR15  ;
2445            /                           & LDMAR           ;
2446   00192    /                           & CMG0 & DREAD    ;  RESOLVE ADDRESS WORD
2447                      JP    INCFTCH     & AM2901 PC,Y,RAMF,PASS,ZA
2448   00193    /                           & SPN0P           ; LOAD Y WITH RETURN ADDRES
2449            ;                                          GO LOAD PC WITH QREG + 1, FET
2450            ;                                        (THE MAR CONTAINS THE FETCH ADDR
2451            ;*****
2452            ;*****    - JPY - Jump indexed by Y
```

```
LINE    ADDR    STATEMENT

2453            ;*****    - PC := Operand Addr + Y
2454            ;*****    - Indirection IS allowed for the operand
2455            ;*****
2456            ;         THIS INSTRUCTION LOOKS FUNNY FOR COMPATIBILITY REASONS
2457            ;
2458            JPY:      CONT              & AM2901 ,TABQ,QREG,ADD,DZ
2459            /                           & CARRYL          ;
2460            /                           & CONDEXT IR2     ; NEVER TRUE
2461            /                           & LDMAR           ;
2462            /                           & CMGO & DREAD    ; FOR COMPATIBILITY
2463   00194    /                           & CON12 H#129     ; FOR COMPATIBILITY (129=IN
2464                      JP    INCPC       & AM2901 Y,PC,RAMF,ADD,AQ
2465            /                           & CARRYL          ;
2466            /                           & LDMAR           ;
2467   00195    /                           & IFETCH          ; COMPUTE ADDR AND FETCH
2468            ;*****
2469            ;*****    - Load A/B indexed by X/Y
2470            ;*****
2471            LABXY:    CCALL INDRES      & AM2901 BR,TABQ,QREG,ADD,DZ
2472            /                           & CARRYL          ;
2473            /                           & CONDEXT MDIR15  ;
2474            /                           & LDMAR           ;
2475   00196    /                           & CMGO & DREAD    ; GET EFFECTIVE ADDR IN Q
2476                      CONT              & AM2901 CXY,,NOP,ADD,AQ
2477            /                           & CARRYL          ;
2478            /                           & LDMAR           ;
2479   00197    /                           & DREAD           ; ADD X/Y AND READ OPERAND
2480                      JP    LD.         & AM2901 PC,PC,RAMA,ADD,ZB
2481            /                           & CARRYH          ;
2482   00198    /                           & LDMAR           ; NEXT INST ADDR
2483            ;*****
2484            ;*****    - LDX. : Load X/Y with (MEM)
2485            ;*****
2486            LDX.:     CALL  READIND     & AM2901 BR,TABQ,QREG,ADD,DZ
2487            /                           & CARRYL          ;
2488            /                           & LODUSR          ;
2489            /                           & LDMAR           ;
2490   00199    /                           & DREAD           ; GET OPERAND
2491                      JP    INCPC       & AM2901 TAB,CXY,RAMF,PASS,DZ
2492   0019A    /                           & IFETCH          ; PUT IT IN X, FETCH
2493            ;*****
2494            ;*****    - Store A/B indexed by X/Y
2495            ;*****
2496            SABXY:    CCALL INDRES      & AM2901 BR,TABQ,QREG,ADD,DZ
2497            /                           & CARRYL          ;
2498            /                           & CONDEXT MDIR15  ;
2499            /                           & LDMAR           ;
2500   0019B    /                           & CMGO & DREAD    ;  GET EFFECTIVE ADDR IN Q
2501                      CONT              & AM2901 CXY,,NOP,ADD,AQ
2502            /                           & CARRYL          ;
2503   0019C    /                           & LDMAR           ; LOAD MAR WITH INDEXED ADD
2504                      JP    SKIP        & AM2901 CAB,TAB,NOP,PASS,ZA
2505            /                           & LDMDOR          ;
2506   0019D    /                           & DWRITE          ; WRITE, FETCH NEXT INSTR.
```

```
LINE    ADDR    STATEMENT

2507            ;*****
2508            ;*****    - STX - Store X/Y to MEM
2509            ;*****
2510            STX.:     CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
2511            /                          & CARRYL          ;
2512            /                          & CONDEXT MDIR15  ;
2513            /                          & LDMAR           ;
2514    0019E   /                          & CMGO & DREAD    ;  RESOLVE ADDRESS WORD
2515                      JP    SKIP       & AM2901 CXY,TAB,NOP,PASS,ZA
2516            /                          & LDMDOR          ;
2517    0019F   /                          & DWRITE          ; WRITE X/Y TO MEMORY
2518            ;*****
2519            ;*****    - Exchange A/B with X/Y
2520            ;*****
2521            XABXY:    CONT             & AM2901 CAB,S0,RAMF,PASS,ZA
2522    001A0   /                          & SPNOP           ; S0  := A/B
2523                      CONT             & AM2901 CXY,CAB,RAMF,PASS,ZA
2524    001A1   /                          & IFETCH          ; A/B := X/Y, FETCH
2525                      JZ               & AM2901 S0,CXY,RAMF,PASS,ZA
2526    001A2   /                          & SPNOP           ; X/4 := S0
```

```
LINE    ADDR    STATEMENT

2528            ;*************************************************************************
2529            ;*                                                                       *
2530            ;*        Bit Manipulation Instructions                                  *
2531            ;*        -----------------------------                                  *
2532            ;*                                                                       *
2533            ;* Format is:   BIT-OPCODE                                               *
2534            ;*              DEF MASK                                                 *
2535            ;*              DEF TARGET                                               *
2536            ;*                                                                       *
2537            ;* Bit opcodes are:                                                      *
2538            ;*        CBS - clear bits                                               *
2539            ;*        SBS - set bits                                                 *
2540            ;*        TBS - test bits                                                *
2541            ;*                                                                       *
2542            ;*************************************************************************
2543            ;
2544            CBS:      CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
2545            /                          & CARRYL         ;
2546            /                          & LODUSR         ;
2547            /                          & LDMAR          ;
2548   001A3    /                          & DREAD          ; GET FIRST OPERAND
2549                      CALL  READRES    & AM2901 TAB,S1,RAMF,EXNOR,DZ
2550   001A4    /                          & CREAD          ; S1    := INVERTED MASK
2551            ;                                             RESOLVE AND READ TARGET
2552                      CONT             & AM2901 ,S1,QREG,PASS,ZB
2553   001A5    /                          & SPNOP          ; QREG := S1
2554                      JP    TWIEXIT    & AM2901 TAB,TAB,NOP,AND,AQ
2555            /                          & LDMDOR         ;
2556   001A6    /                          & DWRITE         ; WRITE TARGET WORD
2557            ;*****
2558            ;*****
2559            ;*****
2560            SBS:      CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
2561            /                          & CARRYL         ;
2562            /                          & LODUSR         ;
2563            /                          & LDMAR          ;
2564   001A7    /                          & DREAD          ; GET OPERANDS
2565                      CALL  READRES    & AM2901 TAB,S1,RAMF,PASS,DZ
2566   001A8    /                          & CREAD          ; S1    := MASK
2567            ;                                             RESOLVE AND READ TARGET
2568                      CONT             & AM2901 ,S1,QREG,PASS,ZB
2569   001A9    /                          & SPNOP          ; QREG := S1
2570                      JP    TWIEXIT    & AM2901 TAB,TAB,NOP,OR,AQ
2571            /                          & LDMDOR         ;
2572   001AA    /                          & DWRITE         ; WRITE TARGET WORD
2573            ;
2574            TBS:      CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
2575            /                          & CARRYL         ;
2576            /                          & LODUSR         ;
2577            /                          & LDMAR          ;
2578   001AB    /                          & DREAD          ; GET OPERANDS
2579                      CALL  READRES    & AM2901 TAB,S1,RAMF,PASS,DZ
2580   001AC    /                          & CREAD          ; S1    := MASK
```

```
LINE    ADDR    STATEMENT

2581            ;                                           RESOLVE AND READ TARGET
2582                    CONT            & AM2901 ,S1,QREG,PASS,ZB
2583    001AD   /                       & SPNOP             ; QREG := S1
2584                    CONT            & AM2901 TAB,S0,RAMF,AND,AQ
2585    001AE   /                       & SPNOP             ; S0 := THE BITS TO TEST
2586                    CONT            & AM2901 S0,,NOP,EXOR,AQ
2587    001AF   /                       & LODUSR            ; TEST THE BITS FOR ALL ONE
2588                    JP      TWIEXIT & AM2901 ,PC,RAMF,ADD,ZB
2589            /                       & CARRYEXT          ;
2590    001B0   /                       & CONDUSR NZ        ; IF NOT ALL ONES, SKIP
```

```
LINE    ADDR    STATEMENT

2592            ;*********************************************************************
2593            ;*                                                                   *
2594            ;*        Byte Manipulation Instructions                             *
2595            ;*        ------------------------------                             *
2596            ;*                                                                   *
2597            ;*      All of these instructions use a byte address.  A byte        *
2598            ;* address is two times the normal address plus 0 or 1 if the        *
2599            ;* high (bits 8-15) or low (bits 0-7) byte is desired.               *
2600            ;*                                                                   *
2601            ;* Notes: S0 contains byte address                                   *
2602            ;*        S5 contains a H#00FF to mask off a byte                    *
2603            ;*        S4 contains byte to load or to store                       *
2604            ;*                                                                   *
2605            ;*********************************************************************
2606            ;*****
2607            LBT:      CONT            & AM2901 B,S0,SRAMR,PASS,ZA
2608   001B1    /                         & SHIFT ROTATE    ;  S0 := WORD ADDRESS
2609                      CONT            & AM2901 BREL,S0,RAMF,ADD,ZB
2610            /                         & CARRYL          ;
2611            /                         & LODMSR          ;
2612            /                         & LDMAR           ; ADD BASE REGISTER IF BASE
2613   001B2    /                         & DREAD           ; START READ OF WORD
2614                      CONT            & AM2901 ,PC,NOP,SUBR,ZB
2615            /                         & CARRYH          ;
2616   001B3    /                         & LDMAR           ; MAR := NEXT INST ADDR
2617                      CCALL BYTESWAP  & AM2901 TAB,S4,RAMF,PASS,DZ
2618   001B4    /                         & CONDMSR NSGN    ;   S4 := WORD;
2619            ;                                             SWAP IF EVEN BYTE ADDRE
2620                      CONT            & AM2901 S4,A,RAMF,NOTRS,DA
2621            /                         & IMM H#FF00      ;
2622   001B5    /                         & IFETCH          ; MASK OFF UPPER BYTE AND F
2623            INCB:     JZ              & AM2901 ,B,RAMF,ADD,ZB
2624   001B6    /                         & CARRYH          ; INCREMENT BYTE ADDRESS IN
2625            ;*****
2626            ;*****
2627            ;*****
2628            SBT:      CONT            & AM2901 B,S0,SRAMR,PASS,ZA
2629   001B7    /                         & SHIFT ROTATE    ;  S0 := WORD ADDRESS
2630                      LDCT  ODBYTE    & AM2901 BREL,S0,RAMF,ADD,ZB
2631            /                         & CARRYL          ;
2632            /                         & LODMSR          ;
2633            /                         & LDMAR           ; ADD BR IF BASE RELATIVE,
2634   001B8    /                         & DREAD           ; START READ OF TARGET WORD
2635                      CONT            & AM2901 A,S4,RAMF,NOTRS,DA
2636   001B9    /                         & IMM H#FF00      ; MASK OFF UPPER BYTE
2637                      JSRP  EVBYTE    & AM2901 TAB,S5,RAMF,PASS,DZ
2638   001BA    /                         & CONDMSR NSGN    ;   S5 := TARGET WORD
2639            ;                                             CALL EVBYTE OR ODBYTE
2640                      CONT            & AM2901 S4,TAB,NOP,OR,AQ
2641            /                         & LDMDOR          ;
2642   001BB    /                         & DWRITE          ; WRITE WORD
2643                      JP    INCB      & AM2901 ,PC,NOP,SUBR,ZB
2644            /                         & CARRYH          ;
```

```
LINE    ADDR    STATEMENT

2645            /                       & LDMAR           ;
2646    001BC   /                       & IFETCH          ; LOAD MAR ; FETCH ; INC B
2647            ODBYTE:   RET           & AM2901 S5,S5,QREG,AND,DA
2648    001BD   /                       & IMM H#FF00      ; MASK TARGET BYTE
2649            EVBYTE:   CONT          & AM2901 S5,S5,QREG,NOTRS,DA
2650    001BE   /                       & IMM H#FF00      ; MASK TARGET BYTE;
2651            ;*****                                      BYTESWAP S4
2652            ;*****    BYTESWAP S4
2653            ;*****
2654            BYTESWAP: CONT          & AM2901 S4,S4,RAMA,PASS,DZ
2655    001BF   /                       & SPRD RL4        ; BYTE SWAP S4
2656            L4MORE:   RET           & AM2901 S4,S4,RAMA,PASS,DZ
2657    001C0   /                       & SPRD RL4        ;
```

```
LINE   ADDR   STATEMENT

2659           ;*****
2660           ;*****   CBT - COMPARE BYTES
2661           ;*****
2662           CBT:     CALL  INITIAL    & AM2901 BR,TABQ,QREG,PASS,DZ
2663           /                         & CARRYL          ;
2664           /                         & LODUSR          ;
2665           /                         & LDMAR           ;
2666   001C1   /                         & DREAD           ; S0 = COUNT
2667                    PUSH             & AM2901 A,S5,SRAMR,PASS,ZA
2668           /                         & LODMSR SWAPEO   ;
2669           /                         & ENBLC & ENBLO   ;
2670   001C2   /                         & SHIFT ROTATE    ;
2671                    CONT             & AM2901 ,S5,NOP,PASS,ZB
2672           /                         & LODMSR          ;
2673           /                         & LDMAR           ;
2674   001C3   /                         & DREAD           ; READ BYTE ARRAY 1
2675                    CONT             & AM2901 B,S5,SRAMR,PASS,ZA
2676   001C4   /                         & SHIFT ROTATE    ;  S5 = BYTE ADDR 2
2677                    CCALL BYTESWAP   & AM2901 TAB,S4,RAMF,PASS,DZ
2678   001C5   /                         & CONDMSR NSGN    ;  IF EVEN BYTE ADDRESS
2679                    CONT             & AM2901 ,S5,NOP,PASS,ZB
2680           /                         & LODMSR          ;
2681           /                         & LDMAR           ;
2682   001C6   /                         & DREAD           ; READ BYTE ARRAY 2
2683                    CONT             & AM2901 S4,S3,RAMF,NOTRS,DA
2684   001C7   /                         & IMM H#FF00      ; S3 = BYTE 1
2685                    CCALL BYTESWAP   & AM2901 TAB,S4,RAMF,PASS,DZ
2686   001C8   /                         & CONDMSR NSGN    ;  IF EVEN BYTE
2687                    CONT             & AM2901 S4,S2,RAMF,NOTRS,DA
2688   001C9   /                         & IMM H#FF00      ; S2 = BYTE 2
2689                    LDCT  CMWNEQ     & AM2901 S2,S3,NOP,SUBR,AB
2690           /                         & CARRYL          ;
2691           /                         & LODMSR          ;  COND REG := BYTE1 - BYTE
2692   001CA   /                         & ENBLO           ;
2693                    JRP   LC05       & AM2901 ,S0,RAMF,SUBR,ZB
2694           /                         & CARRYEXT        ;
2695   001CB   /                         & CONDUSR Z       ; IF STRINGS NOT EQUAL, JR
2696           LC05:    LDCT  CBTDONE    & AM2901 ,A,RAMF,ADD,ZB
2697           /                         & CARRYH          ;
2698   001CC   /                         & LODMSR SWAPUM   ;  INC A ; MSR = COUNT STAT
2699                    CJP   INTCBT     & AM2901 ,B,RAMF,ADD,ZB
2700           /                         & CARRYH          ;
2701   001CD   /                         & CONDEXT INTRPT ; INC B ; IF INT PEND, JP
2702                    LOOP             & AM2901 A,S5,SRAMR,PASS,ZA
2703           /                         & CONDMSR Z       ;
2704   001CE   /                         & SHIFT ROTATE    ;  S1 = WORD ADDRESS
2705           CBTDONE: JP    ENDCBT     & AM2901 PC,A,RAMA,SUBR,ZB
2706           /                         & CARRYH          ;
2707   001CF   /                         & LDMAR           ; DONE: MAR=PC,A=A-1 (FOR E
2708           INTCBT:  JRP   INTCMW     & AM2901 ,S0,RAMF,SUBR,ZB
2709           /                         & CARRYH          ;
2710   001D0   /                         & CONDMSR NZ      ; FIX S0 FOR INTCMW ; IF Z,
```

```
LINE    ADDR    STATEMENT

2712            ;*********************************************************************
2713            ;*                                                                   *
2714            ;*        JSB MBT                                                    *
2715            ;*        DEF COUNT                                                  *
2716            ;*        NOP                                                        *
2717            ;*                                                                   *
2718            ;*        MOVES (COUNT) BYTES FROM A TO B                            *
2719            ;*        COUNT IS A BYTE COUNT >0                                   *
2720            ;*        THE THIRD WORD IS USED TO SAVE THE INTERRUPTED COUNT       *
2721            ;*                                                                   *
2722            ;*********************************************************************
2723            ;
2724            MBT:    CALL  INITIAL   & AM2901 BR,TABQ,QREG,ADD,DZ
2725            /                       & CARRYL         ;
2726            /                       & LODUSR         ;
2727            /                       & LDMAR          ;
2728   001D1    /                       & DREAD          ; S0 := BYTE COUNT (BC)
2729                    LDCT  FILL      & AM2901 A,B,NOP,SUBR,AB
2730            /                       & CARRYH         ; CHECK FOR DEST=SRC+1 CORN
2731   001D2    /                       & LODUSR         ;
2732                    JRP   MBYTE     & AM2901 ,,NOP,ADD,AQ
2733   001D3    /                       & CONDUSR NZ     ; IF Z, FILL
```

```
LINE   ADDR   STATEMENT

2735          ;*********************************************************************
2736          ;*                                                                   *
2737          ;*        SFB - SCAN FOR BYTE                                        *
2738          ;*                                                                   *
2739          ;*********************************************************************
2740          ;
2741          SFB:      CONT            & AM2901 B,S5,SRAMR,PASS,ZA
2742   001D4  /                         & SHIFT ROTATE    ;  S5 = WORD ADDRESS
2743                    CALL  INCS5     & AM2901 BREL,S5,RAMF,ADD,ZB
2744          /                         & CARRYL          ;
2745          /                         & LODMSR          ;
2746          /                         & LDMAR           ;
2747   001D5  /                         & DREAD           ; BREL, READ INITIAL WORD
2748                    CONT            & AM2901 A,S0,RAMF,NOTRS,DA
2749   001D6  /                         & IMM H#FF00      ; S0=TEST BYTE, ODD ADDR
2750                    CONT            & AM2901 A,S1,RAMF,AND,DA
2751   001D7  /                         & IMM H#FF00      ; S1=TERM BYTE, EVEN ADDR
2752                    CALL  L4MORE    & AM2901 A,S4,RAMA,PASS,DZ
2753   001D8  /                         & SPRD RL4        ; S4 = SWAPPED A
2754                    CONT            & AM2901 S4,S2,RAMF,AND,DA
2755   001D9  /                         & IMM H#FF00      ; S2 = TEST BYTE, EVEN ADDR
2756                    CONT            & AM2901 S4,S3,RAMF,NOTRS,DA
2757   001DA  /                         & IMM H#FF00      ; S3 = TERM BYTE, ODD ADDR
2758          SLOOP:    CJP   SODD      & AM2901 TAB,S6,RAMF,PASS,DZ
2759   001DB  /                         & CONDMSR SGN     ; S6=INIT WORD ; JP IF INIT
2760                    CONT            & AM2901 S6,S4,RAMF,AND,DA
2761   001DC  /                         & IMM H#FF00      ; S4 = BYTE TO TEST (EVEN)
2762                    CJP   INTERRPT  & AM2901 S2,S4,NOP,EXOR,AB
2763          /                         & CONDEXT INTRPT ;
2764   001DD  /                         & LUSRCOND        ; COMPARE S2 & S4 ; IF INT,
2765                    CJP   FIXMAR    & AM2901 S1,S4,NOP,EXOR,AB
2766   001DE  /                         & CONDUSR Z       ; COMPARE S1&S4;DONE IF PRE
2767                    CJP   SKIP      & AM2901 S6,B,RAMF,ADD,ZB
2768          /                         & CARRYH          ;
2769   001DF  /                         & CONDUSR Z       ; INC B ; DONE IF PREV COMP
2770          SODD:     CONT            & AM2901 S6,S4,RAMF,NOTRS,DA
2771   001E0  /                         & IMM H#FF00      ; S4 = BYTE TO TEST (ODD)
2772                    CJP   INTERRPT  & AM2901 S0,S4,NOP,EXOR,AB
2773          /                         & CONDEXT INTRPT ;
2774   001E1  /                         & LUSRCOND        ; COMPARE S0&S4 ; IF INT, J
2775                    CJP   FIXMAR    & AM2901 S3,S4,NOP,EXOR,AB
2776   001E2  /                         & CONDUSR Z       ; COMPARE S3&S4;IF PREV COM
2777                    CJP   SKIP      & AM2901 S5,B,RAMA,ADD,ZB
2778          /                         & CARRYH          ; INC B,
2779          /                         & CONDUSR Z       ; IF PREV COMPARE, DONE,
2780          /                         & LDMAR           ; ELSE READ NEXT WORD
2781   001E3  /                         & CNMGO & DREAD   ;
2782                    JP    SLOOP     & AM2901 ,S5,RAMF,ADD,ZB
2783          /                         & CARRYH          ; INC WORD ADDR
2784   001E4  /                         & LODMSR RESET    ; LOOP, CLEAR MSR
```

```
LINE    ADDR    STATEMENT

2786            ;*********************************************************************
2787            ;*                                                                   *
2788            ;*       Word Manipulation Instructions                              *
2789            ;*       ------------------------------                              *
2790            ;*                                                                   *
2791            ;*    These are the old MEF Series interruptible instructions.       *
2792            ;* A zero word must follow the instruction which is used to          *
2793            ;* store a residual count.  The from address is in the A reg,        *
2794            ;* the to address is in the B-reg and the count is pointed to        *
2795            ;* by the second word of the instruction.                            *
2796            ;*                                                                   *
2797            ;*********************************************************************
2798            ;
2799            MVW:      CALL  INITIAL    & AM2901 BR,TABQ,QREG,ADD,DZ
2800            /                          & CARRYL         ;
2801            /                          & LODUSR         ;
2802            /                          & LDMAR          ;
2803   001E5    /                          & DREAD          ; GET THE CORRECT COUNT IN
2804            LMVW:     CJP   INTPEND    & AM2901 A,S0,RAMA,SUBR,ZB
2805            /                          & CARRYH         ; DEC COUNT
2806            /                          & CONDEXT INTRPT ; IF INT PENDING, INT
2807            /                          & LODMSR         ; MSR = COUNT STATUS
2808            /                          & LDMAR          ; MAR = SOURCE ADDRESS
2809   001E6    /                          & CNMGO & DREAD  ; READ SOURCE WORD IF NO IN
2810                      LDCT  LMVW       & AM2901 B,B,RAMA,ADD,ZB
2811            /                          & CARRYH         ; INC B
2812   001E7    /                          & LDMAR          ; MAR := DESTINATION ADDR;
2813                      CONT             & AM2901 TAB,TAB,NOP,PASS,DZ
2814            /                          & LDMDOR         ;
2815   001E8    /                          & DWRITE         ; WRITE DESTINATION WORD;
2816                      JRP   SKIP       & AM2901 ,A,RAMF,ADD,ZB
2817            /                          & CARRYH         ; INC A
2818   001E9    /                          & CONDMSR Z      ; IF COUNT=0, DONE, ELSE LO
2819            ;*****
2820            ;*****    CMW - COMPARE WORDS
2821            ;*****
2822            CMW:      CALL  INITIAL    & AM2901 BR,TABQ,QREG,ADD,DZ
2823            /                          & CARRYL         ;
2824            /                          & LODUSR         ;
2825            /                          & LDMAR          ;
2826   001EA    /                          & DREAD          ; GET THE CORRECT COUNT IN
2827                      CONT             & AM2901  ,A,NOP,PASS,ZB
2828            /                          & LODMSR SWAPEO  ;
2829            /                          & ENBLC & ENBLO  ;
2830            /                          & LDMAR          ;
2831   001EB    /                          & DREAD          ; READ ARRAY 1 WORD
2832            LCMW:     CJP   INTCMW     & AM2901 ,S0,RAMF,SUBR,ZB
2833            /                          & CARRYH         ; DEC COUNT
2834            /                          & CONDEXT INTRPT ; IF INT PEND, JP
2835   001EC    /                          & LODMSR         ; SAVE COUNT STATUS IN MSR
2836                      CONT             & AM2901 B,B,RAMA,ADD,ZB
2837            /                          & CARRYH         ; INC B
2838            /                          & LDMAR          ;
```

```
LINE   ADDR   STATEMENT

2839   001ED  /                          & DREAD          ; READ ARRAY 2 WORD
2840                 CJP   ENDCMW        & AM2901 TAB,,QREG,PASS,DZ
2841   001EE  /                          & CONDMSR Z      ; IF COUNT = 0, JP;
2842          ;                                             QREG := ARRAY 1 WORD
2843                 CONT                & AM2901 TAB,,NOP,SUBR,AQ
2844          /                          & CARRYL         ;
2845          /                          & LODMSR         ; COMPARE THE TWO WORDS.
2846   001EF  /                          & ENBLO          ;     MSR := RESULT OF COMP
2847                 CJP   LCMW          & AM2901 PC,A,RAMF,ADD,ZB
2848          /                          & CARRYEXT       ; IF WORDS EQUAL,
2849          /                          & CONDMSR Z      ;    INC A; READ NEXT WORD;
2850          /                          & LDMAR          ; ELSE FALL THROUGH
2851   001F0  /                          & CMGO & DREAD   ;    (NO INC, NO READ)
2852          CMWNEQ:CONT                & AM2901 PC,PC,RAMF,ADD,DA
2853          /                          & CARRYEXT       ;
2854          /                          & CONDMSR        ; !! CONDMSR SGT !!
2855          /                          & IMM H#0001     ;
2856   001F1  /                          & LDMAR          ; LOAD PC, MAR WITH PROPER
2857                 CONT                & AM2901 S0,B,RAMF,ADD,AB
2858          /                          & CARRYL         ;
2859          /                          & LODMSR SWAPEO  ; RESTORE 0
2860   001F2  /                          & ENBLC & ENBLO  ; B POINTS PAST END OF ARRA
2861          RSTRE: CONT                & AM2901 ,S1,SRAML,PASS,ZB
2862          /                          & SHIFT ROTATEC  ;
2863   001F3  /                          & IFETCH         ; RESTORE E; FETCH NEXT_INS
2864                 JZ                  & AM2901 ,PC,RAMF,ADD,ZB
2865   001F4  /                          & CARRYH         ; INCREMENT PC
2866          ;
2867          ENDCMW:CONT                & AM2901 TAB,,NOP,SUBR,AQ
2868          /                          & CARRYL         ;
2869          /                          & LODMSR         ;  COMPARE THE TWO WORDS
2870   001F5  /                          & ENBLO          ;     MSR := RESULT OF COMP
2871                 CJP   CMWNEQ        & AM2901 ,PC,RAMF,PASS,ZB
2872          /                          & CONDMSR NZ     ;
2873   001F6  /                          & LDMAR          ; LOAD MAR WITH "=" EXIT
2874          ENDCBT:JP     RSTRE        & AM2901 ,A,RAMF,ADD,ZB
2875          /                          & CARRYH         ; INC A
2876          /                          & LODMSR SWAPEO  ; RESTORE 0
2877   001F7  /                          & ENBLC & ENBLO  ;
```

```
LINE    ADDR    STATEMENT

2879            ;
2880            ;       Word Manipulation Subroutines
2881            ;
2882            ;       INITIAL - Initialize S0 with the proper word count
2883            ;
2884            INITIAL:  CCALL MRGIND      & AM2901 PC,S1,SRAMR,PASS,ZA
2885            /                           & CONDUSR SGN     ; !! SHIFT CODE IS 0100 !!
2886   001F8    /                           & LDMAR           ; RESOLVE COUNT; SAVE MC IN
2887                      CONT              & AM2901 TAB,S0,RAMF,PASS,DZ
2888            /                           & LODUSR          ;
2889   001F9    /                           & DREAD           ; S0 := COUNT, READ RESIDUA
2890                      CJP   SKIP        & AM2901 ,PC,RAMF,ADD,ZB
2891            /                           & CARRYH          ;
2892   001FA    /                           & CONDUSR Z       ; INC PC; IF COUNT = 0, EXI
2893                      CONT              & AM2901 TAB,,QREG,PASS,DZ
2894   001FB    /                           & LODUSR          ; QREG := RESIDUAL
2895                      CRET              & AM2901 ,S2,RAMF,AND,ZQ
2896   001FC    /                           & CONDUSR Z       ; S2:=0 ; RETURN IF NO RESI
2897                      RET               & AM2901 S2,S0,RAMA,PASS,ZQ
2898            /                           & LDMDOR          ;
2899   001FD    /                           & DWRITE          ; ZERO RESIDUAL; S0 := COUN
2900            ;
2901            ;       INTPEND - Interrupt pending during block type instruction
2902            ;
2903            INTCMW:   CONT              & AM2901 ,,NOP,ADD,AQ
2904            /                           & LODMSR SWAPEO   ; RESTORE 0
2905   001FE    /                           & ENBLC & ENBLO   ;
2906                      CONT              & AM2901 ,S1,SRAML,PASS,ZB
2907   001FF    /                           & SHIFT ROTATEC   ; RESTORE E
2908            INTPEND:  CONT              & AM2901 ,PC,RAMF,SUBR,ZB
2909            /                           & CARRYH          ;
2910   00200    /                           & LDMAR           ; BACK UP PC
2911                      JP    INTERRPT    & AM2901 ,S0,NOP,ADD,ZB
2912            /                           & CARRYH          ;
2913            /                           & LDMDOR          ;
2914   00201    /                           & DWRITE          ; WRITE RESIDUAL COUNT TO W
```

```
LINE    ADDR    STATEMENT

2916            ;*********************************************************************
2917            ;*                                                                   *
2918            ;*        DOUBLE INTEGER OPERATIONS                                  *
2919            ;*                                                                   *
2920            ;* - Calling sequence:                                               *
2921            ;*                                                                   *
2922            ;*   .DIN, .DDE & .DNG        All others                             *
2923            ;*                                                                   *
2924            ;*     JSB  .OPCODE           JSB  .OPCODE                           *
2925            ;*                            DEF  OPERAND                           *
2926            ;*                                                                   *
2927            ;* - Operands are stored in memory with most significant             *
2928            ;*   word in lower address, least significant word in upper          *
2929            ;*   address:                                                        *
2930            ;*                                                                   *
2931            ;*         JSB  .DAD     +---->OCT   MSW    address                  *
2932            ;*         DEF   *-------+     OCT   LSW    address + 1              *
2933            ;*                                                                   *
2934            ;* - The operations performed are:                                   *
2935            ;*                                                                   *
2936            ;*        .DAD  (A,B) := (A,B) + (OPERAND)                           *
2937            ;*        .DSB  (A,B) := (A,B) - (OPERAND)                           *
2938            ;*        .DSBR (A,B) := (OPERAND) - (A,B)                           *
2939            ;*        .DNG  (A,B) := - (A,B)                                     *
2940            ;*        .DIN  (A,B) := (A,B) + 1                                   *
2941            ;*        .DDE  (A,B) := (A,B) - 1                                   *
2942            ;*        .DIS  (OPERAND) := (OPERAND) + 1, SKIP IF ZERO             *
2943            ;*        .DDS  (OPERAND) := (OPERAND) - 1, SKIP IF ZERO             *
2944            ;*        .DCO  IF (A,B) = (OPERAND) THEN PC := PC + 1               *
2945            ;*              IF (A,B) < (OPERAND) THEN PC := PC + 2               *
2946            ;*              IF (A,B) > (OPERAND) THEN PC := PC + 3               *
2947            ;*                                                                   *
2948            ;* - E Reg can be set but NEVER cleared as follows:                  *
2949            ;*                                                                   *
2950            ;*        .DAD  E set if an unsigned carryout occurs                 *
2951            ;*        .DSB  E set if an unsigned borrow occurs                   *
2952            ;*        .DSBR E set if an unsigned borrow occurs                   *
2953            ;*        .DNG  E set if (A,B) = 0                                   *
2954            ;*        .DIN  E set if (A,B) = -1                                  *
2955            ;*        .DDE  E set if (A,B) = 0                                   *
2956            ;*                                                                   *
2957            ;* - O Reg is cleared and can be set as follows:                     *
2958            ;*                                                                   *
2959            ;*        .DAD  O set if carry into sign XOR carry out of sign       *
2960            ;*        .DSB  O set if carry into sign XOR carry out of sign       *
2961            ;*        .DSBR O set if carry into sign XOR carry out of sign       *
2962            ;*        .DNG  O set if (A,B) = 2**31                               *
2963            ;*        .DIN  O set if (A,B) = 2**31 - 1                           *
2964            ;*        .DDE  O set if (A,B) = 2**31                               *
2965            ;*                                                                   *
2966            ;*********************************************************************
2967           DBLIMSW:  EQU   S3          ; DOUBLE INTEGER SCRATCH REGS
2968           DBLILSW:  EQU   S7
```

```
LINE    ADDR    STATEMENT

2970            ;
2971            ;           .DAD - DOUBLE ADD
2972            ;
2973            .DAD:       CALL  DIARG       & AM2901 BR,TABQ,QREG,ADD,DZ
2974            /                             & CARRYL          ;
2975            /                             & LODUSR          ;
2976            /                             & LDMAR           ;
2977    00202   /                             & DREAD           ; GET OPERAND
2978                        CONT              & AM2901 DBLILSW,B,RAMF,ADD,AB
2979            /                             & CARRYL          ;
2980    00203   /                             & LODUSR          ; ADD LOWER WORDS
2981                        JP    DBLIE       & AM2901 DBLIMSW,A,RAMF,ADD,AB
2982            /                             & CARRYREG        ;
2983            /                             & LODMSR CREG     ;
2984            /                             & ENBLO           ;
2985    00204   /                             & IFETCH          ; ADD UPPER WORDS + CARRY
2986            ;
2987            .DSB:       CALL  DIARG       & AM2901 BR,TABQ,QREG,ADD,DZ
2988            /                             & CARRYL          ;
2989            /                             & LODUSR          ;
2990            /                             & LDMAR           ;
2991    00205   /                             & DREAD           ; GET OPERAND
2992                        CONT              & AM2901 DBLILSW,B,RAMF,SUBR,AB
2993            /                             & CARRYL          ;
2994    00206   /                             & LODUSR          ; B = B-LSW
2995                        CONT              & AM2901 DBLIMSW,A,RAMF,SUBR,AB
2996            /                             & CARRYREG        ;
2997            /                             & LODMSR CI       ;
2998            /                             & ENBLO           ;
2999    00207   /                             & IFETCH          ; A = A - MSW - CARRY
3000            DBLIE:      CJP   FETCH       & AM2901 ,,NOP,ADD,AQ
3001    00208   /                             & CONDUSR NC      ; IF NO CARRY, JP
3002                        JZ                & AM2901 ,,NOP,ADD,AQ
3003            /                             & LODMSR SET      ;
3004    00209   /                             & ENBLC           ;
3005            ;
3006            ;           .DSBR - DOUBLE SUBTRACT REVERSE
3007            ;
3008            ;  - NOTE: Sense of carry is adjusted for DBLIE test
3009            ;
3010            .DSBR:      CALL  DIARG       & AM2901 BR,TABQ,QREG,ADD,DZ
3011            /                             & CARRYL          ;
3012            /                             & LODUSR          ;
3013            /                             & LDMAR           ;
3014    0020A   /                             & DREAD ;         ; GET OPERAND
3015                        CONT              & AM2901 DBLILSW,B,RAMF,SUBS,AB
3016            /                             & CARRYH          ;
3017    0020B   /                             & LODUSRCI        ; B := B - LSW
3018                        JP    DBLIE       & AM2901 DBLIMSW,A,RAMF,SUBS,AB
3019            /                             & CARRYREG        ;
3020            /                             & LODMSR CI       ;
3021            /                             & ENBLO           ;
3022    0020C   /                             & IFETCH          ; A := A - MSW - CARRY
```

```
LINE    ADDR    STATEMENT

3024            ;
3025            ;           .DNG - DOUBLE NEGATE
3026            ;           .DIN - DOUBLE INCREMENT
3027            ;
3028            .DNG:       CONT            & AM2901 ,A,RAMF,EXNOR,ZB
3029    0020D   /                           & SPNOP          ; ONES COMPLEMENT A & B
3030                        CONT            & AM2901 ,B,RAMF,EXNOR,ZB
3031    0020E   /                           & SPNOP          ;
3032            ;
3033            .DIN:       CONT            & AM2901 ,B,RAMF,ADD,ZB
3034            /                           & CARRYH         ;
3035    0020F   /                           & LODUSR         ; B := B + 1
3036                        JP    DBLIE     & AM2901 ,A,RAMF,ADD,ZB
3037            /                           & CARRYREG       ;
3038            /                           & LODMSR CREG    ;
3039            /                           & ENBLO          ;
3040    00210   /                           & IFETCH         ; A := A + CARRY OUT
3041            ;
3042            ;           .DDE - DOUBLE DECREMENT
3043            ;
3044            .DDE:       CONT            & AM2901 ,B,RAMF,SUBR,ZB
3045            /                           & CARRYH         ;
3046    00211   /                           & LODUSR         ; B := B - 1
3047                        JP    DBLIE     & AM2901 ,A,RAMF,SUBR,ZB
3048            /                           & CARRYREG       ;
3049            /                           & LODMSR CI      ;
3050            /                           & ENBLO          ;
3051    00212   /                           & IFETCH         ; A := A - CARRY, UC := BOR
```

```
LINE    ADDR    STATEMENT

3053            ;
3054            ;       DIS - DOUBLE INCREMENT AND SKIP IF ZERO
3055            ;
3056            .DIS:   CALL  DIARG     & AM2901 BR,TABQ,QREG,ADD,DZ
3057            /                       & CARRYL        ;
3058            /                       & LODUSR        ;
3059            /                       & LDMAR         ;
3060    00213   /                       & DREAD         ; GET OPERAND INTO SCRATCH
3061                    CONT            & AM2901 ,DBLILSW,RAMF,ADD,ZB
3062            /                       & CARRYH        ;
3063    00214   /                       & LODUSR        ; INC LSW
3064                    JP    .DSKIPZ   & AM2901 S2,DBLIMSW,RAMA,ADD,ZB
3065            /                       & CARRYUC       ; PROPAGATE CARRY
3066    00215   /                       & LDMAR         ; LDMAR WITH LSW ADDRESS
3067            ;
3068            ;       DDS - DOUBLE DECREMENT AND SKIP IF ZERO
3069            ;
3070            .DDS:   CALL  DIARG     & AM2901 BR,TABQ,QREG,ADD,DZ
3071            /                       & CARRYL        ;
3072            /                       & LODUSR        ;
3073            /                       & LDMAR         ;
3074    00216   /                       & DREAD         ; GET OPERAND INTO SCRATCH
3075                    CONT            & AM2901 ,DBLILSW,RAMF,SUBR,ZB
3076            /                       & CARRYH        ;
3077    00217   /                       & LODUSRCI      ; DEC LSW
3078                    CONT            & AM2901 S2,DBLIMSW,RAMA,SUBR,ZB
3079            /                       & CARRYUC       ; PROPAGATE BORROW
3080    00218   /                       & LDMAR         ; LDMAR WITH LSW ADDR
3081            .DSKIPZ: CONT           & AM2901 DBLILSW,TAB,NOP,PASS,ZA
3082            /                       & LDMDOR        ;
3083    00219   /                       & DWRITE        ; WRITE LSW
3084                    CONT            & AM2901 ,S2,NOP,SUBR,ZB
3085            /                       & CARRYH        ;
3086    0021A   /                       & LDMAR         ; ADDRESS OF MSW
3087                    CONT            & AM2901 DBLIMSW,TAB,NOP,PASS,ZA
3088            /                       & LDMDOR        ;
3089    0021B   /                       & DWRITE        ; WRITE MSW
3090                    CONT            & AM2901 S3,PC,NOP,SUBR,ZB
3091            /                       & CARRYH        ;
3092    0021C   /                       & LDMAR         ; FIX MAR FOR SKIFZ
3093                    JP    SKIFZ     & AM2901 DBLIMSW,DBLILSW,NOP,OR,AB
3094    0021D   /                       & LODUSR        ; SKIP IF ZERO
```

```
LINE    ADDR    STATEMENT

3096            ;*********************************************************************
3097            ;*                                                                   *
3098            ;*      Double Integer Multiply.                                     *
3099            ;*                                                                   *
3100            ;*      JSB .DMP                                                     *
3101            ;*      DEF OPERAND                                                  *
3102            ;*                                                                   *
3103            ;*********************************************************************
3104            ;
3105            .DMP:   CALL  DIARG     & AM2901 BR,TABQ,QREG,ADD,DZ
3106            /                       & CARRYL        ;
3107            /                       & LODUSR        ;
3108            /                       & LDMAR         ;
3109    0021E   /                       & DREAD         ; GET OPERAND
3110                    CONT            & AM2901 B,S0,RAMF,PASS,ZA
3111            /                       & LODMSR RESET  ;
3112    0021F   /                       & ENBLO         ;
3113                    LDCT  H#00F     & AM2901 ,A,NOP,OR,ZB
3114            /                       & CARRYH        ;
3115    00220   /                       & LODMSR        ;
3116                    CJP   DMP1      & AM2901 S7,S3,NOP,OR,ZB
3117            /                       & CARRYH        ;
3118    00221   /                       & CONDUSR UGT   ; NOT ZERO OR ONES
3119            ;
3120            ;       (A,B) is in [-2**16,+2**16).
3121            ;
3122                    CONT            & AM2901 S3,S7,QREG,PASS,ZB
3123    00222   /                       & SPNOP         ;
3124                    JP    DMP2      & AM2901 S3,A,RAMF,PASS,ZA
3125    00223   /                       & SPNOP         ;
3126            ;
3127            ;       (S3,S7) is in [-2**16,+2**16).   (or overflow)
3128            ;
3129            DMP1:   CJP   DMP9      & AM2901 S7,S0,RAMF,PASS,ZA
3130    00224   /                       & CONDUSR UGT   ; IF NOT ZERO OR ONES
3131                    CONT            & AM2901 ,S3,NOP,PASS,ZB
3132    00225   /                       & LODMSR        ;
3133                    CONT            & AM2901 A,S3,RAMF,PASS,ZA
3134    00226   /                       & SPNOP         ;
3135                    CONT            & AM2901 B,S7,RAMF,PASS,ZA
3136    00227   /                       & SPNOP         ;
3137                    CONT            & AM2901 ,B,QREG,PASS,ZB
3138    00228   /                       & SPNOP         ;
3139            ;
3140            ;       At this point:  Mn = multiplier upper (sign only).
3141            ;                       R4 = multiplier lower.
3142            ;                   S3 = A  = multiplicand upper.
3143            ;                   S7 = Q  = multiplicand lower.
3144            ;
3145            ;       Form lower product in (S6,Q)  (Counter = 15)
3146            ;
3147            DMP2:   CPUSH           & AM2901 ,S6,SRAMQR,AND,ZB
3148            /                       & CONDMSR OVR   ; NEVER LOAD COUNTER
```

```
LINE    ADDR    STATEMENT

3149    00229   /                       & SHIFT B#0110    ;
3150                    RFCT            & AM2901 MPY8,S6,SRAMQR,ADD,AB
3151            /                       & CARRYL          ;
3152    0022A   /                       & SHIFT B#1011    ;
3153                    CONT            & AM2901 ,B,RAMF,PASS,ZQ
3154    0022B   /                       & SPNOP           ;
3155            ;
3156            ;       Form upper product in (S4,Q); add middle words.
3157            ;
3158                    LDCT  H#00F     & AM2901 ,S0,QREG,PASS,ZB
3159    0022C   /                       & SPNOP           ;
3160                    CPUSH           & AM2901 ,S4,SRAMQR,AND,ZB
3161            /                       & CONDMSR OVR     ; NEVER LOAD COUNTER
3162    0022D   /                       & SHIFT B#0000    ; Load Q0BUF.
3163                    RFCT            & AM2901 MPY,S4,SRAMQR,ADD,AB
3164            /                       & CARRYL          ;
3165    0022E   /                       & SHIFT B#1110    ;
3166                    CONT            & AM2901 S6,A,RAMF,ADD,AQ
3167            /                       & CARRYL          ;
3168    0022F   /                       & LODUSR          ;
3169                    CONT            & AM2901 S7,S4,RAMF,ADD,ZB
3170    00230   /                       & CARRYUC         ;
3171            ;
3172            ;       If multiplier upper is -1, subtract multiplicand.
3173            ;
3174                    CJP   DMP3      & AM2901 S7,,NOP,ADD,AQ
3175    00231   /                       & CONDMSR NSGN    ;
3176                    CONT            & AM2901 S7,A,RAMF,SUBR,AB
3177            /                       & CARRYL          ;
3178    00232   /                       & LODUSR          ;
3179                    CONT            & AM2901 S3,S4,RAMF,SUBR,AB
3180    00233   /                       & CARRYNUC        ;
3181            ;
3182            ;       Check for overflow:  upper 17 bits must match.
3183            ;
3184            DMP3:   CONT            & AM2901 ,S4,NOP,OR,ZB
3185            /                       & CARRYH          ;
3186    00234   /                       & LODUSR          ;
3187                    CJP   DMP9      & AM2901 A,S4,NOP,EXOR,AB
3188    00235   /                       & CONDUSR UGT     ; IF NOT ZERO OR ONES
3189                    CJP   FETCH     & AM2901 ,,NOP,ADD,AQ
3190            /                       & CONDUSR NSGN    ;
3191    00236   /                       & CMGO & IFETCH   ;
3192            ;
3193            ;       Overflow.  Return (77777,177777) and overflow set.
3194            ;
3195            DMP9:   CONT            & AM2901 A,A,SRAMR,EXNOR,AB
3196            /                       & IFETCH          ;
3197    00237   /                       & SHIFT B#0000    ;
3198                    JZ              & AM2901 B,B,RAMF,EXNOR,AB
3199            /                       & LODMSR SET      ;
3200    00238   /                       & ENBLO           ;
```

```
LINE    ADDR    STATEMENT

3202            ;*********************************************************************
3203            ;*                                                                   *
3204            ;*       Double integer divide. (And divide reverse)                 *
3205            ;*                                                                   *
3206            ;*       JSB .DDI (.DDIR)                                            *
3207            ;*       DEF OPERAND                                                 *
3208            ;*                                                                   *
3209            ;*********************************************************************
3210            ;
3211            ;       Get operands.  For .DDIR, swap them.
3212            ;
3213            .DDI:   CALL  DIARG     & AM2901 BR,TABQ,QREG,ADD,DZ
3214            /                       & CARRYL         ;
3215            /                       & LODUSR         ;
3216            /                       & LDMAR          ;
3217   00239    /                       & DREAD          ;
3218                    JP    DDI01     & AM2901 A,S5,RAMF,PASS,ZA
3219   0023A    /                       & LODUSR         ;
3220            .DDIR:  CALL  DIARG     & AM2901 BR,TABQ,QREG,ADD,DZ
3221            /                       & CARRYL         ;
3222            /                       & LODUSR         ;
3223            /                       & LDMAR          ;
3224   0023B    /                       & DREAD          ;
3225                    CONT            & AM2901 S7,S5,RAMF,PASS,ZA
3226   0023C    /                       & SPNOP          ;
3227                    CONT            & AM2901 B,S7,RAMF,PASS,ZA
3228   0023D    /                       & SPNOP          ;
3229                    CONT            & AM2901 S5,B,RAMF,PASS,ZA
3230   0023E    /                       & SPNOP          ; SWAPERANDS
3231                    CONT            & AM2901 S3,S5,RAMF,PASS,ZA
3232   0023F    /                       & SPNOP          ;
3233                    CONT            & AM2901 A,S3,RAMF,PASS,ZA
3234   00240    /                       & SPNOP          ;
3235                    CONT            & AM2901 S5,A,RAMF,PASS,ZA
3236   00241    /                       & LODUSR         ;
3237            ;
3238            ;       Take absolute value of operands.  R9 = sign diff.
3239            ;
3240            DDI01:  CJP   DDI02     & AM2901 S3,S5,RAMF,EXOR,AB
3241   00242    /                       & CONDUSR NSGN   ;
3242                    CONT            & AM2901 ,B,RAMF,SUBS,ZB
3243            /                       & CARRYH         ;
3244   00243    /                       & LODUSR         ;
3245                    CONT            & AM2901 ,A,RAMF,SUBS,ZB
3246   00244    /                       & CARRYUC        ;
3247            DDI02:  CONT            & AM2901 ,S3,NOP,PASS,ZB
3248   00245    /                       & LODUSR         ;
3249                    CJP   DDI03     & AM2901 ,S3,NOP,PASS,ZB
3250   00246    /                       & CONDUSR NSGN   ;
3251                    CONT            & AM2901 ,S7,RAMF,SUBS,ZB
3252            /                       & CARRYH         ;
3253   00247    /                       & LODUSR         ;
3254                    CONT            & AM2901 ,S3,RAMF,SUBS,ZB
```

```
LINE    ADDR    STATEMENT

3255    00248   /                       & CARRYUC       ;
3256            DDI03:  CJP   DDI04     & AM2901 ,S7,NOP,PASS,ZB
3257    00249   /                       & CONDUSR NZ    ;
3258                    CJP   DMP9      & AM2901 ,S7,NOP,PASS,ZB
3259    0024A   /                       & CONDUSR Z     ; Divide by zero.
3260                    CJP   DDI04     & AM2901 S7,A,QREG,PASS,ZB
3261    0024B   /                       & CONDUSR SGN   ;
3262            ;
3263            ;       Divisor is in [1,32767].  Use two 2/1 divides.
3264            ;       First divide generates 15 bits, which are put in A.
3265            ;       Second generates 17 bits, which are put in A<0> & B.
3266            ;
3267                    PUSH  H#00F     & AM2901 S7,S0,RAMF,AND,ZQ
3268    0024C   /                       & DIVCOND       ;
3269                    RFCT            & AM2901 S7,DIV,SRAMQL,SUBR,AB
3270            /                       & CARRYL        ;
3271            /                       & DIVUCOND      ;
3272    0024D   /                       & SHIFT B#1111  ;
3273                    CONT            & AM2901 ,A,RAMF,PASS,ZQ
3274    0024E   /                       & LODMSR SWAPUM ;
3275                    PUSH  H#010     & AM2901 S7,B,QREG,PASS,ZB
3276    0024F   /                       & CONDMSR SGN   ;
3277                    RFCT            & AM2901 S7,DIV,SRAMQL,SUBR,AB
3278            /                       & CARRYL        ;
3279            /                       & DIVUCOND      ;
3280    00250   /                       & SHIFT B#1111  ;
3281                    CONT            & AM2901 ,B,RAMF,EXNOR,ZQ
3282    00251   /                       & LODMSR SWAPUM ; (IS THE LODMSR REALLY NEC
3283                    CONT            & AM2901 ,S0,SRAMQR,PASS,ZB
3284    00252   /                       & SHIFT B#1111  ;
3285                    JP    DDI06     & AM2901 ,A,SRAMQL,PASS,ZB
3286    00253   /                       & SHIFT B#1111  ;
```

```
LINE    ADDR    STATEMENT

3288            ;       Divisor > 32767.  Use one 3/2 divide loop, with
3289            ;       (S4,S0,Q) / (S3,S7).  S6 has .not.S3, so that an
3290            ;       ADD is used instead of a SUBR, which has the wrong
3291            ;       carry-in;  CARRYNUC would conflict with DIVCOND.
3292            ;       Note that Mc is saved in Mo across this loop.
3293            ;
3294            DDI04:  CONT            & AM2901 ,S4,RAMF,AND,ZQ
3295            /                       & LODMSR SWAPEO  ; Save Mc
3296   00254    /                       & ENBLC & ENBLO  ;
3297                    CONT            & AM2901 A,S0,RAMF,PASS,ZA
3298            /                       & LODMSR RESET   ; Mc, Mn = 0
3299   00255    /                       & ENBLC          ;
3300                    CONT            & AM2901 ,B,QREG,PASS,ZB
3301   00256    /                       & SPNOP          ;
3302                    PUSH  H#010     & AM2901 S3,S6,RAMF,EXNOR,ZA
3303   00257    /                       & CONDMSR C      ;
3304                    CJP   DDI05     & AM2901 S7,DIV,SRAMQL,SUBR,AB
3305            /                       & CARRYL         ;
3306            /                       & CONDMSR C      ;
3307   00258    /                       & SHIFT B#1100   ;
3308                    RFCT            & AM2901 S6,S4,SRAML,ADD,AB
3309            /                       & UCDIVCND       ;
3310   00259    /                       & SHIFT B#1001   ;
3311                    CONT            & AM2901 A,A,SRAMQL,EXNOR,AB
3312   0025A    /                       & SHIFT B#1100   ;
3313                    JP    DDI06     & AM2901 ,B,RAMF,EXNOR,ZQ
3314            /                       & LODMSR SWAPEO  ;
3315   0025B    /                       & ENBLC & ENBLO  ;
3316            ;
3317            ;            *** NOTE ***
3318            ;       DDI05 MUST ALIGN WITH 4 LSBITS = B#1100
3319            ;
3320            DDI05:  RFCT            & AM2901 S3,S4,SRAML,ADD,AB
3321            /                       & UCDIVCND       ;
3322   0025C    /                       & SHIFT B#1001   ;
3323                    CONT            & AM2901 A,A,SRAMQL,EXNOR,AB
3324   0025D    /                       & SHIFT B#1100   ;
3325                    CONT            & AM2901 ,B,RAMF,EXNOR,ZQ
3326            /                       & LODMSR SWAPEO  ;
3327   0025E    /                       & ENBLC & ENBLO  ;
```

```
LINE    ADDR    STATEMENT

3329            ;       If operand signs differed, negate result.
3330            ;       If not, check for overflow (-2**31 / -1).
3331            ;
3332            DDI06:  LDCT  DMP9      & AM2901 ,S5,NOP,ADD,ZB
3333            /                       & CARRYL         ;
3334            /                       & LODMSR         ; Mo=0
3335    0025F   /                       & ENBLO          ;
3336                    CJP   DDI08     & AM2901 ,A,RAMF,EXNOR,ZB
3337    00260   /                       & CONDMSR NSGN   ;
3338                    CONT            & AM2901 ,B,RAMF,SUBS,ZB
3339            /                       & CARRYH         ;
3340            /                       & LODUSR         ;
3341    00261   /                       & IFETCH         ;
3342                    JZ              & AM2901 ,A,RAMF,SUBS,ZB
3343    00262   /                       & CARRYUC        ;
3344            DDI08:  JRP   FETCH     & AM2901 ,,NOP,ADD,AQ
3345            /                       & CONDUSR NSGN   ;
3346    00263   /                       & CMGO & IFETCH  ;
```

```
LINE    ADDR    STATEMENT

3348            ;*********************************************************************
3349            ;*                                                                   *
3350            ;*        .DCO - DOUBLE INTEGER ARITH COMPARE                        *
3351            ;*                                                                   *
3352            ;*        JSB    .DCO                                                *
3353            ;*        DEF    OPERAND                                             *
3354            ;*        JMP    EQUAL              IF (A,B) == (OPERAND)            *
3355            ;*        JMP    LESS_THAN          IF (A,B) <  (OPERAND)            *
3356            ;*        JMP    GREATER_THAN       IF (A,B) >  (OPERAND)            *
3357            ;*                                                                   *
3358            ;*                                                                   *
3359            ;*  - Both operands are considered 32 bit 2's complement numbers     *
3360            ;*                                                                   *
3361            ;*  - Algorithm is do a signed compare of upper words.  If equal     *
3362            ;*    then do an unsigned compare of lower words.                    *
3363            ;*                                                                   *
3364            ;*********************************************************************
3365            .DCO:     CALL  DIARG      & AM2901 BR,TABQ,QREG,ADD,DZ
3366            /                          & CARRYL          ;
3367            /                          & LODUSR          ;
3368            /                          & LDMAR           ;
3369   00264    /                          & DREAD           ;
3370                      CJP   .DCOEQ     & AM2901 A,DBLIMSW,NOP,SUBS,AB
3371            /                          & CARRYH          ;
3372            /                          & CONDUSR Z       ;
3373   00265    /                          & SEQFRZ          ; IF EQUAL, TEST LOWER BITS
3374            .DCOLT:   JP    INCPC      & AM2901 PC,PC,RAMF,ADD,ZB
3375            /                          & CARRYEXT        ;
3376            /                          & CONDUSR SGT     ;
3377            /                          & LDMAR           ;
3378   00266    /                          & IFETCH          ; IF GREATER THAN, INC PC
3379            ;
3380            ;         DO UNSIGNED COMPARE OF LOWER WORDS
3381            ;
3382            .DCOEQ:   CJP   SKIFNZ     & AM2901 B,DBLILSW,NOP,SUBS,AB
3383            /                          & CARRYH          ; IF LSW'S COMPARE, JP TO
3384            /                          & CONDUSR Z       ; SKIFNZ, WHICH WILL *NOT*
3385   00267    /                          & SEQFRZ          ; SKIP, BUT WILL IFETCH
3386                      JP    INCPC      & AM2901 PC,PC,RAMF,ADD,ZB
3387            /                          & CARRYEXT        ;
3388            /                          & CONDUSR UGT     ;
3389            /                          & LDMAR           ;
3390   00268    /                          & IFETCH          ; ADD ONE TO PC IF GREATER
```

```
LINE    ADDR    STATEMENT

3392            ;*********************************************************************
3393            ;*                                                                   *
3394            ;*        DIARG - DOUBLE INTEGER ARGUMENT LOAD                       *
3395            ;*                                                                   *
3396            ;*        - Operand is loaded into DBLIMSW and DBLILSW reg pair      *
3397            ;*        - Assumes first argument read in progress                  *
3398            ;*                                                                   *
3399            ;*********************************************************************
3400            DIARG:    CCALL READRES     & AM2901 ,,NOP,ADD,AQ
3401   00269    /                           & CONDUSR SGN     ; RESOLVE OPERAND
3402                      CONT              & AM2901 ,S2,RAMF,ADD,ZQ
3403            /                           & CARRYH          ;
3404   0026A    /                           & LDMAR           ; S2=MAR= LSW ADDRESS
3405                      CALL FIXPC        & AM2901 TAB,DBLIMSW,RAMF,PASS,DZ
3406   0026B    /                           & DREAD           ; DBLIMSW := MSW
3407                      RET               & AM2901 TAB,DBLILSW,RAMF,PASS,DZ
3408   0026C    /                           & SPNOP           ; DBLILSW := LSW
```

```
LINE   ADDR   STATEMENT

3410          ;*********************************************************************
3411          ;*                                                                   *
3412          ;*        .SETP - PUT (A:=A+1) IN (B:=B+1) (COUNT) TIMES, WHERE      *
3413          ;*                COUNT IS THE SECOND WORD OF THE INSTRUCTION        *
3414          ;*                COUNT IS *NOT* A/B ADDRESSABLE                     *
3415          ;*                                                                   *
3416          ;*        ON RESTART FROM INTERRUPT, A CONTAINS THE RESIDUAL         *
3417          ;*        COUNT (MINUS 1) , AND B CONTAINS THE RESIDUAL ADDRESS      *
3418          ;*        WITH BIT 15 SET.                                           *
3419          ;*                                                                   *
3420          ;*********************************************************************
3421          ;
3422          .SETP:    CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
3423          /                          & CARRYL          ;
3424          /                          & LODUSR          ;
3425          /                          & LDMAR           ;
3426  0026D   /                          & DREAD           ; GET COUNT
3427                    CONT             & AM2901 B,B,SRAMR,ADD,AB
3428          /                          & CARRYL          ;
3429          /                          & LODUSR          ;
3430  0026E   /                          & SHIFT B#0000    ; UC := BIT 15, CLEAR BIT 1
3431                    CJP   INTD       & AM2901 BREL,B,RAMF,ADD,ZB
3432          /                          & CARRYL          ; IF BASE RELATIVE, ADD BR
3433          /                          & CONDUSR C       ; JP IF RESTARTING FROM INT
3434  0026F   /                          & LDMAR           ;
3435                    LDCT             & AM2901 ,S4,RAMF,SUBS,DZ
3436          /                          & CARRYL          ;
3437          /                          & LUSRCOND        ;
3438  00270   /                          & SPETC LDCTY     ; R4=COUNT-1; CNTR=LOW NIBB
3439                    CJP   SKIP       & AM2901 B,S4,QREG,ADD,AB
3440          /                          & CARRYH          ;
3441  00271   /                          & CONDUSR SGN     ; DONE IF NEGATIVE; Q=TERMI
3442          SETPL:    CJP   SETPINT    & AM2901 A,A,RAMA,ADD,ZB
3443          /                          & CARRYH          ;
3444          /                          & CONDEXT INTRPT  ;
3445          /                          & LDMDOR          ;
3446  00272   /                          & DWRITE          ; WRITE TABLE ENTRY ; IF IN
3447                    RPCT  SETPL      & AM2901 ,B,RAMF,ADD,ZB
3448          /                          & CARRYH          ;
3449  00273   /                          & LDMAR           ; LOOP 'TIL DONE
3450                    LDCT             & AM2901 S4,S4,RAMF,SUBR,DA
3451          /                          & CARRYH          ;
3452          /                          & IMM H#000F      ;
3453  00274   /                          & LUSRCOND        ; DO THE NEXT 16 TABLE ENTR
3454                    CJP   SETPL      & AM2901 PC,PC,NOP,ADD,AQ
3455  00275   /                          & CONDUSR NSGN    ; CHECK TO SEE IF DONE
3456                    JZ               & AM2901 PC,PC,RAMA,ADD,ZB
3457          /                          & CARRYH          ;
3458          /                          & LDMAR           ;
3459  00276   /                          & IFETCH          ;
3460          ;
3461          INTD:     CONT             & AM2901 B,B,RAMA,ADD,ZB
3462          /                          & CARRYH          ;
```

```
LINE    ADDR    STATEMENT

3463            /                           & LDMAR           ;
3464    00277   /                           & DREAD           ; READ OLD A VALUE
3465                    LDCT                & AM2901 A,S4,RAMF,SUBR,ZA
3466            /                           & CARRYH          ;
3467            /                           & LUSRCOND        ;
3468    00278   /                           & SPETC LDCTY     ; R4=CNTR=USR=RESIDUAL COUN
3469                    CJP     SKIP        & AM2901 B,A,RAMA,ADD,DZ
3470            /                           & CARRYH          ;
3471            /                           & CONDUSR SGN     ;
3472    00279   /                           & LDMAR           ; RESTORE A;IF COUNT 0, SKI
3473                    JP      SETPL       & AM2901 B,S4,QREG,ADD,AB
3474    0027A   /                           & CARRYH          ; Q=TERMINAL B; START LOOPI
3475            ;
3476            SETPINT:LDCT    INTERRPT    & AM2901 B,A,RAMF,SUBR,AQ
3477            /                           & CARRYH          ;
3478    0027B   /                           & LODMSR SET      ;
3479                    JRP                 & AM2901 B,B,SRAMR,ADD,AB
3480            /                           & CARRYH          ;
3481            /                           & CONDMSR NSGN    ;
3482    0027C   /                           & SHIFT ROTATE    ; SET SIGN OF B; ALWAYS JR
```

```
LINE    ADDR    STATEMENT

3484            ;*********************************************************************
3485            ;*                                                                   *
3486            ;*         .CPM - SINGLE INTEGER ARITH COMPARE                       *
3487            ;*                                                                   *
3488            ;*         JSB    .CPM                                               *
3489            ;* MAR -> DEF    OP1[,I]           - DEF's may reference A/B        *
3490            ;*  PC -> DEF    OP2[,I]                                            *
3491            ;*         JMP    EQUAL             - IF OP1 == OP2                  *
3492            ;*         JMP    LESS_THAN         - IF OP1 <  OP2                  *
3493            ;*         JMP    GREATER_THAN      - IF OP1 >  OP2                  *
3494            ;*                                                                   *
3495            ;*       - Both operands are considered signed 16 bit numbers        *
3496            ;*                                                                   *
3497            ;*********************************************************************
3498            .CPM:     CALL   READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
3499            /                           & CARRYL          ;
3500            /                           & LODUSR          ;
3501            /                           & LDMAR           ;
3502   0027D    /                           & DREAD           ; RESOLVE OP1
3503                      CALL   READRES    & AM2901 TAB,S0,RAMF,PASS,DZ
3504   0027E    /                           & CREAD           ; RESOLVE OP2
3505                      CONT              & AM2901 ,S0,QREG,PASS,ZB
3506   0027F    /                           & SPNOP           ; Q = OP1
3507            ;
3508            ;         QREG := (OP1) AND TAB := (OP2)
3509            ;
3510                      CONT              & AM2901 ,PC,RAMF,ADD,ZB
3511            /                           & CARRYH          ;
3512   00280    /                           & LDMAR ;         ; MAR IS '=' EXIT
3513                      CONT              & AM2901 TAB,,NOP,SUBS,AQ
3514            /                           & CARRYH          ;
3515   00281    /                           & LODMSR ;        ; COMPARE WORDS
3516                      CJP    SKIFNZ     & AM2901 ,PC,RAMF,ADD,ZB
3517            /                           & CARRYH          ; LOAD USR FROM PREV LINE
3518   00282    /                           & LUSRSP          ; COND IS SGE
3519                      JP     INCPC      & AM2901 ,PC,RAMF,ADD,ZB
3520            /                           & CARRYH          ;
3521            /                           & LDMAR           ;
3522   00283    /                           & IFETCH          ; OP1 > OP2 : ADD 1 TO PC,
```

```
LINE    ADDR    STATEMENT

3524            ;*********************************************************************
3525            ;                                                                    *
3526            ;         DFER/CFER/ZFER/XFER.                                       *
3527            ;                                                                    *
3528            ;*********************************************************************
3529            ;
3530            ;         READ 1ST DEF & RESOLVE;  Q:=ADDR
3531            ;         S1 := Q; MAR:=PC
3532            ;         READ 2ND DEF & RESOLVE; Q:=ADDR.
3533            ;         A := Q
3534            ;         PC := PC+2
3535            ;
3536            DCZFER:   CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
3537            /                          & CARRYL          ;
3538            /                          & CONDEXT MDIR15  ;
3539            /                          & LDMAR           ;
3540   00284    /                          & CMGO & DREAD    ;  IF FIRST DEF INDIRECT
3541                      CALL  READRES    & AM2901 PC,S1,RAMA,PASS,ZQ
3542            /                          & LDMAR           ; S1 = DESTINATION ADDRESS
3543   00285    /                          & CREAD           ; READ SECOND DEF
3544                      CONT             & AM2901 S1,A,RAMA,ADD,ZQ
3545            /                          & CARRYH          ;
3546   00286    /                          & LDMAR           ; A := SOURCE ADDRESS + 1
3547                      CONT             & AM2901 TAB,,NOP,PASS,DZ
3548            /                          & LDMDOR          ;
3549   00287    /                          & DWRITE          ; WRITE FIRST WORD
3550                      CONT             & AM2901 S1,B,RAMF,ADD,ZA
3551   00288    /                          & CARRYH          ; B := DEST ADDRESS + 1
3552                      CONT             & AM2901 PC,PC,RAMF,ADD,DA
3553            /                          & CARRYL          ;
3554   00289    /                          & IMM H#0002      ; PC=PC+2
3555            ;
3556            ;         CNTR := COUNT-2, DECODED FROM INSTRUCTION.
3557            ;           READ FROM (A); A:=A+1
3558            ;           MAR:=B; B:=B+1
3559            ;           WRITE DATA JUST READ.
3560            ;           LOOP DONE (CNTR+1) TIMES.
3561            ;
3562            XFCONT:   PUSH             & AM2901 ,,NOP,ADD,AQ
3563   0028A    /                          & JTAB WORDCNT    ;
3564                      CONT             & AM2901 A,A,RAMA,ADD,ZB
3565            /                          & CARRYH          ;
3566            /                          & LDMAR           ;
3567   0028B    /                          & DREAD           ;
3568                      CONT             & AM2901 B,B,RAMA,ADD,ZB
3569            /                          & CARRYH          ;
3570            /                          & LODMSR RESET    ;
3571   0028C    /                          & LDMAR           ;
3572                      TWB   FIXMAR     & AM2901 TAB,TAB,NOP,PASS,DZ
3573            /                          & CONDMSR SGN     ;
3574            /                          & LDMDOR          ;
3575   0028D    /                          & DWRITE          ; TWB NEVER FALLS THRU
```

```
LINE    ADDR    STATEMENT

3577            ;*********************************************************************
3578            ;*                                                                   *
3579            ;*        .ENTR - RTE Parameter Passing Routine                      *
3580            ;*                                                                   *
3581            ;*  - Calling Sequence:                                              *
3582            ;*                                      PARMS  EQU  *     <----+     *
3583            ;*        JSB   SUB                             BSS  N           |     *
3584            ;*   +--> DEF   *+M+1  ------+          SUB    NOP  ------+     |     *
3585            ;*   |    DEF   P[1]         |                 JSB  .ENTR |     |     *
3586            ;*   |    ...                |   MAR --->   DEF  PARMS | ---+     *
3587            ;*   |    DEF   P[M]         |   PC ---->              |          *
3588            ;*   |    <RETURN ADDR>  <---+                         |          *
3589            ;*   +-------------------------------------------------+          *
3590            ;*                                                                   *
3591            ;*  - Operation:                                                     *
3592            ;*                                                                   *
3593            ;*      - Each DEF is resolved to a true address and moved to        *
3594            ;*        the parameter block before the subroutine.                 *
3595            ;*                                                                   *
3596            ;*      - The actual return address is stored in SUB entry           *
3597            ;*                                                                   *
3598            ;*      - If M >= N then N parameters are passed.                    *
3599            ;*                                                                   *
3600            ;*      - If M < N then M parameters are passed.                     *
3601            ;*                                                                   *
3602            ;*      - M or N can be zero.                                        *
3603            ;*                                                                   *
3604            ;*********************************************************************
3605            ;
3606            ;*****
3607            ;*****    .ENTN - MOVE N PARAMETERS
3608            ;*****
3609            .ENTN:    CONT            & AM2901 PC,S1,RAMF,SUBR,DA
3610            /                         & CARRYL          ;
3611            /                         & IMM H#0003      ;
3612            /                         & LDMAR           ;
3613   0028E    /                         & DREAD ;         ; S1=SUB ; READ SUB
3614                      CONT            & AM2901 TAB,,QREG,PASS,DZ
3615   0028F    /                         & SPNOP ;         ; Q=PARMS
3616                      CONT            & AM2901 S1,S0,RAMF,SUBS,AQ
3617            /                         & CARRYH          ;
3618   00290    /                         & LODUSR ;        ; S0 = COUNT
3619                      CJP   SKIP      & AM2901 S0,,NOP,ADD,DA
3620            /                         & CARRYL          ;
3621            /                         & CONDUSR Z       ;
3622            /                         & LDMDOR          ;
3623   00291    /                         & DWRITE          ; WRITE TRUE RETURN ADDRESS
3624            .ENTN05:  JP    .ENTR0    & AM2901 TAB,S3,RAMF,PASS,DZ
3625            /                         & LODMSR RESET    ; MSR = 0 FOR .INTR
3626            /                         & LDMAR           ;
3627   00292    /                         & DREAD           ; S3=FROM ; READ FIRST FROM
3628            ;*****
3629            ;*****    .ENTC - MOVE N PARAMETERS, RE-ENTRANT AND POSSIBLY PRIVILEDGE
```

```
LINE    ADDR    STATEMENT

3630            ;*****
3631            .ENTC:    CONT              & AM2901 PC,S1,RAMF,SUBR,DA
3632            /                           & CARRYL          ;
3633            /                           & IMM H#0005      ;
3634            /                           & LDMAR           ;
3635    00293   /                           & DREAD           ; S1=SUB ; READ SUB
3636                      LDCT   .ENTN05    & AM2901 TAB,,QREG,PASS,DZ
3637    00294   /                           & SPNOP           ; Q=PARMS ; MSR = 1
3638                      CONT              & AM2901 S1,S0,RAMF,SUBS,AQ
3639            /                           & CARRYH          ;
3640    00295   /                           & LODUSR          ; S0 = COUNT
3641                      JRP    SKIP       & AM2901 S0,A,RAMF,ADD,DA
3642            /                           & CARRYL          ;
3643            /                           & CONDUSR Z       ;
3644            /                           & LDMDOR          ; WRITE TRUE RETURN ADDRESS
3645    00296   /                           & DWRITE          ; IN A AND IN SUB
3646            ;*****
3647            ;*****    .ENTP - MOVE M OR N PARAMETERS, RE-ENTRANT, POSSIBLY PRIVILED
3648            ;*****
3649            .ENTP:    CONT              & AM2901 PC,S1,RAMF,SUBR,DA
3650            /                           & CARRYL          ;
3651            /                           & IMM H#0005      ;
3652            /                           & LDMAR           ;
3653    00297   /                           & DREAD           ; READ SUB ; S1=SUB
3654                      CALL   .ENTRSUB   & AM2901 TAB,,QREG,PASS,DZ
3655    00298   /                           & SPNOP           ; Q = PARMS = TO
3656            ;                                             ;  .ENTRSUB : S0=COUNT,S3=F
3657                      JP     .ENTR0     & AM2901 ,A,RAMF,PASS,DZ
3658    00299   /                           & SPNOP           ; A= TRUE RETURN ADDRESS
3659            ;*****
3660            ;*****    .ENTR - MOVE M OR N PARAMETERS
3661            ;*****
3662            .ENTR:    CONT              & AM2901 PC,S1,RAMF,SUBR,DA
3663            /                           & CARRYL          ;
3664            /                           & IMM H#0003      ;
3665            /                           & LDMAR           ;
3666    0029A   /                           & DREAD           ; READ SUB;S1=SUB
3667                      CALL   .ENTRSUB   & AM2901 TAB,,QREG,PASS,DZ
3668    0029B   /                           & SPNOP           ; Q = PARMS = TO
3669            ;                                             ;  .ENTRSUB : S0=COUNT, S3=
3670            .ENTR0:   CONT              & AM2901 ,S5,RAMF,PASS,ZQ
3671    0029C   /                           & LDMAR           ; S5 = MAR = PARMS
3672            .ENTRL:   CJP    WINDR      & AM2901 TAB,,NOP,PASS,DZ
3673            /                           & CONDEXT MDIR15 ; SPIND DOES LDMAR,DREAD IF
3674    0029D   /                           & SPIND           ;  OR LDMDOR,DWRITE IF NOT
3675                      CONT              & AM2901 ,S0,RAMF,SUBR,ZB
3676            /                           & CARRYH          ;
3677    0029E   /                           & LODUSR          ; DEC COUNT; TEST FOR 0
3678                      CJP    SKIP       & AM2901 ,S3,RAMF,ADD,ZB
3679            /                           & CARRYH          ;
3680            /                           & CONDUSR Z       ;
3681            /                           & LDMAR           ;
3682    0029F   /                           & CNMGO & DREAD   ; INC FROM;MAR=NXT FROM; SK
3683            .ENTRL5:  JP     .ENTRL     & AM2901 ,,QREG,ADD,ZQ
```

```
LINE    ADDR    STATEMENT

3684            /                           & CARRYH          ;
3685    002A0   /                           & LDMAR           ; INC TO ; MAR=TO ; LOOP
3686            WINDR:  CJP   .ENTRL        & AM2901 S5,S6,NOP,PASS,ZQ
3687            /                           & CONDEXT NOABIN ;
3688    002A1   /                           & LDMAR           ; MAR = TO ; CHECK FOR INTE
3689                    CJP   .INTR         & AM2901 S5,S6,RAMF,SUBR,AQ
3690            /                           & CARRYL          ;
3691    002A2   /                           & CONDEXT INTRPT ; S5 = NUMBER MOVED SO FAR
3692                    CONT                & AM2901 TAB,,NOP,PASS,DZ
3693            /                           & LDMDOR          ;
3694    002A3   /                           & DWRITE          ; PUT A/B IN MEMORY
3695                    JP    .ENTRL5       & AM2901 ,,QREG,SUBR,ZQ
3696            /                           & CARRYH          ;
3697    002A4   /                           & DREAD           ; GET A/B IN THE MDIR; FIX
3698            ;
3699            .INTR:  CONT                & AM2901 S6,S1,NOP,PASS,ZB
3700    002A5   /                           & LDMAR           ; MAR = SUB
3701                    JP    INTERRPT      & AM2901 S6,S3,NOP,SUBR,AB
3702            /                           & CARRYEXT        ;
3703            /                           & CONDMSR SGN     ;
3704            /                           & LDMDOR          ;
3705    002A6   /                           & DWRITE          ; RESTORE ORIGINAL SUB ENTR
3706            ;
3707            ;       .ENTRSUB - SETUP ROUTINE FOR .ENTR, .ENTP
3708            ;
3709            .ENTRSUB: CONT              & AM2901 TAB,S3,RAMF,PASS,DZ
3710            /                           & LODMSR SET      ; MSR SET FOR .INTR
3711            /                           & LDMAR           ; S3 = (SUB) = TO - 1
3712    002A7   /                           & DREAD           ; READ DEF TO TRUE RETURN A
3713                    CONT                & AM2901 S1,S2,RAMA,SUBS,AQ
3714            /                           & CARRYH          ;
3715            /                           & LODUSR          ;
3716    002A8   /                           & LDMAR           ; S2 = N ; MAR = SUB
3717                    CJP   SKIP          & AM2901 S3,S0,RAMF,PASS,DZ
3718            /                           & CONDUSR Z       ;
3719            /                           & LDMDOR          ;
3720    002A9   /                           & DWRITE          ; S0 WILL BE M; WRITE TRUE
3721                    CONT                & AM2901 S3,S0,RAMF,SUBR,AB
3722            /                           & CARRYH          ;
3723    002AA   /                           & LODUSR          ; S0 = TRUE RET - (TO-1) -1
3724                    CJP   SKIP          & AM2901 S2,S0,NOP,SUBR,AB
3725            /                           & CARRYL          ;
3726    002AB   /                           & CONDUSR Z       ; COMPARE M AND N
3727                    CRET                & AM2901 S2,S3,RAMF,ADD,ZB
3728            /                           & CARRYH          ;
3729            /                           & CONDUSR SGN     ;
3730            /                           & LDMAR           ;
3731    002AC   /                           & CMGO & DREAD    ; MAR=FROM=S3+1
3732                    RET                 & AM2901 S2,S0,RAMF,PASS,ZA
3733    002AD   /                           & DREAD           ; S0=N
```

```
LINE    ADDR    STATEMENT

3735            ;*****************************************************************
3736            ;                                                               *
3737            ;       '..FCM'  NEGATE FLOATING-POINT VALUE IN (A,B).          *
3738            ;                                                               *
3739            ;       UNDERFLOW AND OVERFLOW CAN OCCUR.                       *
3740            ;                                                               *
3741            ;*****************************************************************
3742            ;
3743            ;       TEST (A)
3744            ;       PC := PC - 1
3745            ;       IF A=0, DONE
3746            ;
3747            ..FCM:  CONT            & AM2901 ,A,NOP,PASS,ZB
3748    002AE   /                       & LODUSR        ;
3749                    CJP   CLRO      & AM2901 ,,NOP,ADD,AQ
3750            /                       & CONDUSR Z     ;
3751    002AF   /                       & CMGO & IFETCH ;
3752            ;
3753            ;       UNPACK (A,B) TO (XU,XL,XEXP)    (XU=A,XL=B)
3754            ;       B := - B
3755            ;       A := - A - BORROW
3756            ;       GOTO ROUND0
3757            ;
3758                    CALL  UNPACK1   & AM2901 B,XEXP,RAMF,PASS,ZA
3759            /                       & LODMSR SWAPEO ;
3760    002B0   /                       & ENBLC & ENBLO ;
3761                    CONT            & AM2901 ,B,RAMF,SUBS,ZB
3762            /                       & CARRYH        ;
3763    002B1   /                       & LODUSR        ;
3764                    JP    ROUND0    & AM2901 ,A,RAMF,SUBS,ZB
3765    002B2   /                       & CARRYUC       ;
```



```
LINE    ADDR    STATEMENT

3767            ;*****************************************************************
3768            ;*                                                               *
3769            ;*   .FLUN - Unpack a single precision floating-point number.    *
3770            ;*                                                               *
3771            ;*****************************************************************
3772            ;
3773            .FLUN:  CONT            & AM2901 B,A,RAMF,NOTRS,DA
3774    002B3   /                       & IMM H#FF00    ; A = EXP | EXP SIGN
3775                    CONT            & AM2901 B,,NOP,AND,DA
3776            /                       & IMM H#0001    ;
3777    002B4   /                       & LUSRCOND      ; USR = EXP SIGN
3778                    CJP   FLUN1     & AM2901 A,B,RAMF,EXOR,AB
3779            /                       & CONDUSR Z     ;
3780    002B5   /                       & IFETCH        ; FETCH NEXT INST
3781                    JZ              & AM2901 A,A,SRAMR,OR,DA
3782    002B6   /                       & IMM H#FF01    ; SIGN EXTEND (Shift = 0001
3783            FLUN1:  JZ              & AM2901 ,A,SRAMR,PASS,ZB
3784    002B7   /                       & SHIFT B#0000  ; A = EXP
```

```
LINE   ADDR   STATEMENT

3786          ;*********************************************************************
3787          ;*                                                                   *
3788          ;*   .PACK - Pack, normalize & round a single-precision number.      *
3789          ;*           The mantissa is contained in the A and B registers.     *
3790          ;*           The exponent is contained in the word following         *
3791          ;*           the .PACK instruction.                                  *
3792          ;*                                                                   *
3793          ;*********************************************************************
3794          ;
3795          .PACK:    CALL  FIXPC      & AM2901 TAB,XEXP,RAMF,PASS,DZ
3796  002B8   /                          & SPNOP          ; XEXP = SECOND WORD
3797                    CONT             & AM2901 XEXP,XEXP,NOP,ADD,AB
3798          /                          & CARRYL         ;
3799  002B9   /                          & LODUSR         ;
3800                    CJP   FOFLUFL    & AM2901 ,XEXP,NOP,PASS,ZB
3801  002BA   /                          & CONDUSR OVR    ;
3802          ;*****
3803          ;*****
3804          ;*****    NORMALIZE.  ENTER WITH (XU,XL) = VALUE;  XEXP = EXPONENT.
3805          ;*****
3806          ;*****    EXIT TO NEXT INSTRUCTION WITH (A,B) = RESULT, 0 SET/RESET.
3807          ;*****
3808          ;*****    Mn := A<15>
3809          ;*****    Q  := XL
3810          ;*****    YEXP := XEXP + 1
3811          ;*****    IF XU=0 THEN
3812          ;*****      B := 0
3813          ;*****      IF XL=0 THEN EXIT
3814          ;*****    XEXP := YEXP + 1
3815          ;*****
3816          NORM:     LDCT  DNCOUNT1   & AM2901 A,S4,RAMF,OR,ZA
3817          /                          & CARRYH         ;
3818          /                          & LODMSR SWAPEO  ;
3819  002BB   /                          & ENBLC & ENBLO  ;
3820                    CJP   NORM1      & AM2901 ,B,QREG,PASS,ZB
3821  002BC   /                          & CONDUSR UGT    ; IF A <> 0 OR 177777, JP
3822                    CONT             & AM2901 XU,XL,NOP,OR,AB
3823  002BD   /                          & LODMSR         ;
3824                    CJP   NORMEND    & AM2901 B,A,RAMF,PASS,ZA
3825  002BE   /                          & CONDUSR Z      ; IF ZEROS, DONE
3826                    CONT             & AM2901 XEXP,XEXP,RAMF,SUBR,DA
3827          /                          & CARRYL         ;
3828  002BF   /                          & IMM H#0010     ; ADJUST EXPONENT
3829                    CONT             & AM2901 A,S4,NOP,EXOR,AB
3830  002C0   /                          & LODUSR         ; CHECK DID NOT DISCARD ALL
3831                    CJP   NOSIGNS    & AM2901 ,,QREG,AND,ZQ
3832  002C1   /                          & CONDUSR SGN    ; JP IF ALL SIGNS DISCARDED
3833          NORM1:    CPUSH            & AM2901 ,A,SRAMQL,PASS,ZB
3834          /                          & CONDMSR Z      ; ALWAYS FAILS
3835  002C2   /                          & SHIFT B#0100   ;
3836                    TWB              & AM2901 ,A,SRAMQL,PASS,ZB
3837          /                          & SHIFT B#0100   ;
3838  002C3   /                          & CONDEXT IRSKBF ;
```

```
LINE    ADDR    STATEMENT

3839                    CONT            & AM2901 ,A,SRAMQR,PASS,ZB
3840            /                       & LODMSR INVERT  ;
3841            /                       & ENBLC          ;
3842    002C4   /                       & SHIFT B#0100   ;
3843                    JSRP            & AM2901 ,A,SRAMQR,PASS,ZB
3844            /                       & CONDMSR NZ     ;
3845    002C5   /                       & SHIFT B#0100   ;
3846                    JP    INCEXP    & AM2901 S7,XEXP,RAMF,SUBR,AB
3847            /                       & CARRYL         ; CORRECT EXPONENT
3848            /                       & LODMSR RESET   ;
3849    002C6   /                       & ENBLC          ; MC = 0 (= MO AFTER SWAP)
3850            NOSIGNS:CONT            & AM2901 S4,S4,NOP,ADD,AB
3851            /                       & CARRYL         ;
3852            /                       & LODMSR         ;
3853    002C7   /                       & ENBLC          ; MC = SIGN
3854                    CONT            & AM2901 ,A,SRAMQR,PASS,ZB
3855    002C8   /                       & SHIFT B#0100   ; SHIFT IT IN
3856            INCEXP: JP    ROUND     & AM2901 ,XEXP,RAMF,ADD,ZB
3857            /                       & CARRYH         ; INC XEXP
3858            /                       & LODMSR SWAPEO  ;
3859    002C9   /                       & ENBLC & ENBLO  ;
3860            ;
3861            NORMEND:JP    CLRO      & AM2901 ,,NOP,ADD,AQ
3862            /                       & LODMSR SWAPEO  ;
3863            /                       & ENBLC & ENBLO  ;
3864    002CA   /                       & IFETCH         ;
3865            ;
3866            ;       ROUND:
3867            ;
3868            ;       YL := 177B
3869            ;       B := Q + YL  (+1 IF A>=0)
3870            ;       A := A + CARRY
3871            ;
3872            ROUND:  CONT            & AM2901 ,YL,RAMF,PASS,DZ
3873    002CB   /                       & IMM H#007F     ;
3874                    CONT            & AM2901 A,A,NOP,ADD,AB
3875            /                       & CARRYL         ;
3876    002CC   /                       & LODUSR         ;
3877                    CONT            & AM2901 YL,B,RAMF,ADD,AQ
3878    002CD   /                       & CARRYNUC       ;
3879                    CONT            & AM2901 ,A,RAMF,ADD,ZB
3880    002CE   /                       & CARRYNUC       ;
```

```
LINE    ADDR    STATEMENT

3882            ;       SPECIAL CASES:
3883            ;
3884            ;       IF OVERFLOW ON CARRY PROPAGATE,
3885            ;         A := A RS 1
3886            ;         XEXP := XEXP + 1
3887            ;       ELSE IF A=140000,
3888            ;         A := A LS 1
3889            ;         XEXP := XEXP - 1
3890            ;
3891            ROUND0: CJP   ROUND1     & AM2901 A,A,NOP,ADD,AB
3892            /                        & CARRYL          ;
3893   002CF    /                        & CONDUSR NOVR    ;
3894                    CONT             & AM2901 ,A,SRAMR,PASS,ZB
3895   002D0    /                        & SHIFT B#0000    ;
3896                    JP    ROUND2     & AM2901 ,XEXP,RAMF,ADD,ZB
3897   002D1    /                        & CARRYH          ;
3898            ROUND1: CJP   ROUND2     & AM2901 ,,NOP,ADD,AQ
3899   002D2    /                        & CONDUSR OVR     ;
3900                    CONT             & AM2901 ,A,SRAML,PASS,ZB
3901   002D3    /                        & SHIFT B#0010    ;
3902                    CONT             & AM2901 ,XEXP,RAMF,SUBR,ZB
3903   002D4    /                        & CARRYH          ;
3904            ;       CHECK FOR EXPONENT UNDERFLOW OR OVERFLOW:
3905            ;       UPPER 9 BITS MUST BE THE SAME.  FORMAT EXPONENT.
3906            ;
3907            ;       YEXP := XEXP .AND. 200B
3908            ;       YEXP := XEXP + YEXP
3909            ;       YEXP := YEXP + YEXP + CARRY
3910            ;       IF (YEXP.AND.177400B) .NE. 0, OVERFLOW/UNDERFLOW
3911            ;
3912            ROUND2: CONT             & AM2901 XEXP,YEXP,RAMF,AND,DA
3913   002D5    /                        & IMM H#0080      ;
3914                    CONT             & AM2901 XEXP,YEXP,RAMF,ADD,AB
3915            /                        & CARRYL          ;
3916   002D6    /                        & LODUSR          ;
3917                    CONT             & AM2901 YEXP,YEXP,RAMF,ADD,AB
3918   002D7    /                        & CARRYUC         ;
3919                    CONT             & AM2901 YEXP,,NOP,AND,DA
3920            /                        & LUSRCOND        ;
3921   002D8    /                        & IMM H#FF00      ;
3922                    CJP   FOFLUFL    & AM2901 ,XEXP,NOP,PASS,ZB
3923   002D9    /                        & CONDUSR NZ      ;
3924            ;
3925            ;       PACK EXPONENT WITH 2ND MANTISSA WORD, CLEAR OVERFLOW, & EXIT.
3926            ;
3927                    CONT             & AM2901 B,B,RAMF,AND,DA
3928            /                        & IMM H#FF00      ;
3929   002DA    /                        & IFETCH          ;
3930                    JZ               & AM2901 YEXP,B,RAMF,OR,AB
3931            /                        & LODMSR RESET    ;
3932   002DB    /                        & ENBLO           ;
3933            ;
```

```
LINE    ADDR    STATEMENT

3935            ;         FLOATING UNDERFLOW & OVERFLOW.
3936            ;         TEST SIGN OF EXPONENT.
3937            ;         NEGATIVE: UNDERFLOW, RESULT = 0
3938            ;         POSITIVE: OVERFLOW, RESULT = 77777B,177776B
3939            ;
3940            FOFLUFL:  CJP   FOFL      & AM2901 ,A,RAMF,AND,ZB
3941            /                         & CONDUSR NSGN   ;
3942   002DC    /                         & CNMGO & IFETCH ;
3943                      JZ              & AM2901 ,B,RAMF,AND,ZB
3944            /                         & LODMSR SET     ;
3945   002DD    /                         & ENBLO          ;
3946            FOFL:     CONT            & AM2901 A,A,SRAMR,EXNOR,AB
3947            /                         & SHIFT B#0000   ; A = 077777B
3948   002DE    /                         & IFETCH         ;
3949                      JZ              & AM2901 B,B,SRAML,EXNOR,AB
3950            /                         & LODMSR SET     ;
3951            /                         & ENBLO          ; SET 0
3952   002DF    /                         & SHIFT B#0010   ; B=177776B
```

```
LINE    ADDR    STATEMENT

3954            ;*********************************************************************
3955            ;*                                                                   *
3956            ;*        .PWR2 - Multiply a real number by a power of 2.            *
3957            ;*                No overflow or underflow checks.                   *
3958            ;*                                                                   *
3959            ;*                THE POWER OF 2 IS NOT A/B ADDRESSABLE              *
3960            ;*                                                                   *
3961            ;*********************************************************************
3962            ;
3963            .PWR2:    CCALL READRES    & AM2901 BR,TABQ,QREG,ADD,DZ
3964            /                          & CARRYL          ;
3965            /                          & CONDEXT MDIR15  ;
3966            /                          & LDMAR           ;
3967    002E0   /                          & DREAD           ; RESOLVE POWER OF 2
3968                      CONT             & AM2901 A,B,NOP,OR,AB
3969    002E1   /                          & LODUSR          ; CHECK FOR ZERO
3970                      CJP    INCPC     & AM2901 PC,S0,RAMA,PASS,DZ
3971            /                          & CONDUSR Z       ; S0 = POWER OF 2
3972            /                          & LDMAR           ; MAR = PC
3973    002E2   /                          & CMGO & IFETCH   ; IF ZERO, DONE
3974                      CONT             & AM2901 B,S1,RAMF,AND,DA
3975    002E3   /                          & IMM H#FF00      ; S1 = MANTISSA
3976                      CONT             & AM2901 S1,B,SRAMQR,EXOR,AB
3977    002E4   /                          & SHIFT B#0110    ; B = EXP, Q=EXP SIGN
3978                      CONT             & AM2901 ,,NOP,PASS,ZQ
3979    002E5   /                          & LODUSR          ; USR = EXP SIGN
3980                      CJP    PWR2POS   & AM2901 S0,B,RAMF,ADD,AB
3981            /                          & CARRYL          ; ADD EXP AND POWER OF 2
3982    002E6   /                          & CONDUSR NSGN    ; IF NEG EXP, MUST ADD SIGN
3983                      CONT             & AM2901 B,B,RAMF,ADD,DA
3984            /                          & CARRYL          ;
3985    002E7   /                          & IMM H#FF80      ; ADD SIGN EXTENSION
3986            PWR2POS:  CONT             & AM2901 ,B,SRAML,PASS,ZB
3987    002E8   /                          & SHIFT B#1010    ; REFORMAT EXPONENT
3988                      CONT             & AM2901 B,B,RAMF,NOTRS,DA
3989    002E9   /                          & IMM H#FF00      ; MASK TO 8 BITS
3990                      JP     INCPC     & AM2901 S1,B,RAMF,OR,AB
3991    002EA   /                          & IFETCH          ; MERGE MANTISSA AND EXP, F
```

```
LINE    ADDR    STATEMENT

3993            ;*********************************************************************
3994            ;*                                                                   *
3995            ;*       Convert single precision to double precision.               *
3996            ;*                                                                   *
3997            ;*       JSB .BLE                                                    *
3998            ;*       DEF RTNADDRESS                                              *
3999            ;*       DEF DPRESULT                                                *
4000            ;*       DEF SPARGUMENT                                              *
4001            ;*                                                                   *
4002            ;*********************************************************************
4003            ;
4004            .BLE:     CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
4005            /                          & CARRYL          ;
4006            /                          & CONDEXT MDIR15  ;
4007            /                          & LDMAR           ;
4008   002EB    /                          & CMG0 & DREAD    ;
4009                      CONT             & AM2901 ,PC,NOP,ADD,ZB
4010            /                          & CARRYH          ;
4011            /                          & LDMAR           ;
4012   002EC    /                          & CREAD           ; GET ARGUMENT
4013                      CONT             & AM2901 ,S5,RAMF,PASS,ZQ
4014   002ED    /                          & SPNOP           ; S5 = RETURN ADDRESS
4015                      CCALL READRES    & AM2901 BR,TABQ,QREG,ADD,DZ
4016            /                          & CARRYL          ;
4017            /                          & CONDEXT MDIR15  ;
4018            /                          & LDMAR           ;
4019   002EE    /                          & DREAD           ; RESOLVE INDIRECTS
4020                      CONT             & AM2901 ,,NOP,ADD,ZQ
4021            /                          & CARRYH          ;
4022   002EF    /                          & LDMAR           ; SECOND WORD ADDRESS
4023                      CONT             & AM2901 TAB,S0,RAMF,PASS,DZ
4024   002F0    /                          & DREAD           ; S0=FIRST WORD, READ 2ND
4025                      CONT             & AM2901 PC,S2,RAMA,AND,ZB
4026   002F1    /                          & LDMAR           ; S2 = 0
4027                      CONT             & AM2901 TAB,S1,RAMF,PASS,DZ
4028   002F2    /                          & CREAD           ; S1 = SECOND WORD
4029                      CONT             & AM2901 S1,S3,RAMF,NOTRS,DA
4030   002F3    /                          & IMM H#FF00      ; S3 = EXP
4031                      CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
4032            /                          & CARRYL          ;
4033            /                          & CONDEXT MDIR15  ;
4034            /                          & LDMAR           ;
4035   002F4    /                          & CMG0 & DREAD    ;
4036                      JP    DSTORE1    & AM2901 S3,S1,RAMF,EXOR,AB
4037            /                          & LODMSR RESET    ;
4038   002F5    /                          & ENBLO           ; S1 = MANTISSA BITS
```

```
LINE    ADDR    STATEMENT

4040            ;*********************************************************************
4041            ;*                                                                   *
4042            ;*       Convert double precision to single precision.               *
4043            ;*                                                                   *
4044            ;*       JSB .NGL                                                    *
4045            ;*       DEF ARGUMENT                                                *
4046            ;*                                                                   *
4047            ;*********************************************************************
4048            ;
4049            ;       Get argument.  Pack into (A,Q,XEXP) and let
4050            ;       single precision finish it.
4051            ;
4052            .NGL:   CCALL INDRES    & AM2901 BR,TABQ,QREG,ADD,DZ
4053            /                       & CARRYL        ;
4054            /                       & CONDEXT MDIR15 ;
4055            /                       & LDMAR         ;
4056   002F6    /                       & CMGO & DREAD  ; RESOLVE RETURN ADDRESS
4057                    CALL  DBLARG1   & AM2901 PC,S5,RAMA,PASS,ZQ
4058            /                       & LDMAR         ; GET ARGS
4059   002F7    /                       & CREAD         ; S5 = RTN ADDRESS
4060                    CONT            & AM2901 S5,PC,RAMA,ADD,ZA
4061            /                       & CARRYH        ;
4062            /                       & LODMSR SWAPEO ;
4063            /                       & ENBLC & ENBLO ;
4064   002F8    /                       & LDMAR         ; MAR = NEXTINST, FIX PC
4065                    CONT            & AM2901 S0,A,RAMF,PASS,ZA
4066   002F9    /                       & LODUSR        ; A = FIRST WORD
4067                    CJP   NGL02     & AM2901 S3,S2,NOP,OR,AB
4068            /                       & CONDUSR Z     ; IF A IS ZERO, DONE
4069   002FA    /                       & CMGO & IFETCH ;
4070                    CJP   NGL01     & AM2901 ,S1,QREG,PASS,ZB
4071   002FB    /                       & CONDUSR Z     ;
4072                    CONT            & AM2901 ,,QREG,OR,DQ
4073   002FC    /                       & IMM H#0001    ;
4074            NGL01:  JP    ROUND     & AM2901 MP,XEXP,RAMF,PASS,ZA
4075   002FD    /                       & SPNOP         ;
4076            NGL02:  JZ              & AM2901 ,B,RAMF,AND,ZQ
4077            /                       & LODMSR RESET  ;
4078   002FE    /                       & ENBLO         ;
```

```
LINE    ADDR    STATEMENT

4080            ;*****************************************************************
4081            ;*                                                               *
4082            ;*       Double precision negate in place.                       *
4083            ;*                                                               *
4084            ;*       JSB ..TCM                                               *
4085            ;*       DEF ARGUMENT                                            *
4086            ;*                                                               *
4087            ;*****************************************************************
4088            ;
4089            ..TCM:    CCALL READRES    & AM2901 BR,TABQ,QREG,ADD,DZ
4090            /                          & CARRYL         ;
4091            /                          & CONDEXT MDIR15 ;
4092            /                          & LDMAR          ;
4093   002FF    /                          & DREAD          ; RESOLVE DEF, READ 1ST WOR
4094                      CALL  DBLARG1A   & AM2901 ,,QREG,ADD,ZQ
4095            /                          & CARRYH         ; GET ARGS
4096            /                          & LODMSR SWAPEO  ;
4097            /                          & ENBLC & ENBLO  ; SWAP E & O
4098   00300    /                          & LDMAR          ; MAR = SECOND WORD ADDRESS
4099                      CALL  DNEGATE    & AM2901 ,S3,RAMF,SUBS,ZB
4100           /                           & CARRYH         ;
4101   00301   /                          & LODUSR         ;
4102                      JP    DPACK1     & AM2901 PC,S5,RAMF,PASS,ZA
4103   00302    /                         & SPNOP          ; S5 POINTS TO NEXT INST
```

```
LINE    ADDR    STATEMENT

4105            ;*********************************************************************
4106            ;*                                                                   *
4107            ;*        OPERATING SYSTEM SET                                       *
4108            ;*                                                                   *
4109            ;*   105300 - .CPUID - Processor Identification                      *
4110            ;*   105301 - .FWID  - Microcode Identification                      *
4111            ;*   105302 - .WFI   - Wait for Interrupt                            *
4112            ;*   105303 - .SIP   - Skip if Interrupt Pending                     *
4113            ;*                                                                   *
4114            ;*********************************************************************
4115            ;*****
4116            ;*****    CPUID - LOAD A REG WITH CPU ID
4117            ;*****
4118            .CPUID:   JZ              & AM2901 ,A,RAMF,PASS,DZ
4119            /                         & IMM CPUID      ;
4120   00303    /                         & IFETCH         ;
4121            ;*****
4122            ;*****    FWID - LOAD A REG WITH MICROCODE ID
4123            ;*****
4124            .FWID:    JZ              & AM2901 ,A,RAMF,PASS,DZ
4125            /                         & IMM MICREVID   ;
4126   00304    /                         & IFETCH         ;
4127            ;*****
4128            ;*****    SIP - SKIP IF I/O INTERRUPT PENDING
4129            ;*****
4130            .SIP:     JZ              & AM2901 PC,PC,RAMA,ADD,ZB
4131            /                         & CARRYEXT       ;
4132            /                         & CONDEXT SINTRQ ;
4133            /                         & CLD & LDMAR    ;
4134   00305    /                         & IFETCH         ; SKIP IF PENDING
4135            ;*****
4136            ;*****    WFI - WAIT FOR INTERRUPT
4137            ;*****
4138            .WFI:     LDCT  .WFI2     & AM2901 ,,NOP,ADD,AQ
4139   00306    /                         & SPNOP          ;
4140            .WFI2:    JRP   INTERRPT  & AM2901 ,,NOP,ADD,AQ
4141   00307    /                         & CONDEXT INTRPT ;
```

```
LINE    ADDR    STATEMENT

4143            ;*********************************************************************
4144            ;*                                                                   *
4145            ;*       DMS Map Feature                                             *
4146            ;*       ---------------                                             *
4147            ;*                                                                   *
4148            ;*  - There are 32 sets of 32 map registers.  The map set is         *
4149            ;*    selected by the Address Extention Reg.  The actual map reg     *
4150            ;*    is selected by bits 10 to 14 of the MAR.  The AER is           *
4151            ;*    loaded from bits 0-5 of the Y bus.  Ex:                        *
4152            ;*                                                                   *
4153            ;*    +-----+-----+-----+-----+-----+-----+                          *
4154            ;*    | ROM |      Map Set Number         |  AER                    *
4155            ;*    +-----+-----+-----+-----+-----+-----+                          *
4156            ;*                                                                   *
4157            ;*    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+              *
4158            ;*    |XX| Map Reg Num  |          Offset              |  MAR        *
4159            ;*    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+              *
4160            ;*                                                                   *
4161            ;*  - The Macro machine can access 3 map sets directly.  They        *
4162            ;*    are called Execute map, Data 1 map, and Data 2 map.            *
4163            ;*    These maps are kept in a 4 word by 8 bit register file         *
4164            ;*    external to the 2901                                           *
4165            ;*                                                                   *
4166            ;*    +--+--+--+--+--+--+--+--+                                      *
4167            ;*    |       Data 2 Map       |  MAPD2                              *
4168            ;*    +--+--+--+--+--+--+--+--+                                      *
4169            ;*                                                                   *
4170            ;*    +--+--+--+--+--+--+--+--+                                      *
4171            ;*    |       Data 1 Map       |  MAPD1                              *
4172            ;*    +--+--+--+--+--+--+--+--+                                      *
4173            ;*                                                                   *
4174            ;*    +--+--+--+--+--+--+--+--+                                      *
4175            ;*    |       Exec Map         |  MAPX                               *
4176            ;*    +--+--+--+--+--+--+--+--+                                      *
4177            ;*                                                                   *
4178            ;*                                                                   *
4179            ;*  - The MAPD1, MAPD2, and MAPX registers are loaded from WMAP      *
4180            ;*    or the Working MAP set.  WMAP has the format:                  *
4181            ;*                                                                   *
4182            ;*    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+              *
4183            ;*    |MP|    DATA2     |    DATA1     |    EXEC      |  WMAP       *
4184            ;*    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+              *
4185            ;*                                                                   *
4186            ;*    Where: MP    - Memory Protect enable                           *
4187            ;*           DATA2 - Data 2 map number                               *
4188            ;*           DATA1 - Data 1 map number                               *
4189            ;*           EXEC  - Execute map number                              *
4190            ;*                                                                   *
4191            ;*********************************************************************
```

```
LINE    ADDR    STATEMENT

4193            ;*********************************************************************
4194            ;*                                                                   *
4195            ;*        Priviledged DMS Instructions                               *
4196            ;*        ----------------------------                               *
4197            ;*                                                                   *
4198            ;*   - The following DMS instructions are priviledged:               *
4199            ;*                                                                   *
4200            ;*        LPMR - Load Page Mapping Register                          *
4201            ;*        SPMR - Store Page Mapping Register                         *
4202            ;*        LDMP - Load Map                                            *
4203            ;*        STMP - Store Map                                           *
4204            ;*        LWD1 - Load Data 1 map number                              *
4205            ;*        LWD2 - Load Data 2 map number                              *
4206            ;*        SWMP - Store Working Map set                               *
4207            ;*        SIMP - Store Interrupt Map set                             *
4208            ;*        XJMP - Cross map JMP                                       *
4209            ;*                                                                   *
4210            ;*   - If any of these instructions are executed while memory        *
4211            ;*     protect is enabled, a memory protect violation will be        *
4212            ;*     generated by the firmware.  After the instruction is          *
4213            ;*     decoded, a test is made for Memory Protect enable.  If        *
4214            ;*     the MP system is on, generate a protect violation.            *
4215            ;*                                                                   *
4216            ;*********************************************************************
```

```
LINE    ADDR    STATEMENT

4218            ;*************************************************************************
4219            ;*                                                                       *
4220            ;*        XL.1 or XL.2                                                   *
4221            ;*        DEF  ADDRESS                                                   *
4222            ;*                                                                       *
4223            ;*        CROSS LOAD A/B FROM ALTERNATE MAP                              *
4224            ;*        INDIRECTS ARE RESOLVED IN THE EXECUTE MAP                      *
4225            ;*        THE FINAL REFERENCE TAKES PLACE WITH BOOT                      *
4226            ;*        MEMORY AND A/B ADDRESSABILITY TURNED OFF                       *
4227            ;*                                                                       *
4228            ;*************************************************************************
4229            ;*****
4230            XL.10:    CCALL INDMS       & AM2901 TAB,,QREG,PASS,DZ
4231            /                           & CONDEXT MDIR15 ;
4232    00308   /                           & LDMAR          ;
4233                      CONT              & AM2901 MAPD1,,NOP,PASS,DZ
4234            /                           & LDAER          ;
4235    00309   /                           & DREAD          ; READ FROM DATA1 MAP
4236            XLCONT:   CONT              & AM2901 MAPX,,NOP,PASS,DZ
4237    0030A   /                           & LDAER          ; PUT EXECUTE MAP BACK IN A
4238                      CONT              & AM2901 PC,PC,RAMA,ADD,ZB
4239            /                           & CARRYH         ;
4240            /                           & LDMAR          ;
4241    0030B   /                           & IFETCH         ;
4242                      JZ                & AM2901 ,CAB,RAMF,PASS,DZ
4243    0030C   /                           & SPNOP          ; LOAD A/B WITH DATA
4244            XL.20:    CCALL INDMS       & AM2901 TAB,,QREG,PASS,DZ
4245            /                           & CONDEXT MDIR15 ;
4246    0030D   /                           & LDMAR          ;
4247                      JP    XLCONT      & AM2901 MAPD2,,NOP,PASS,DZ
4248            /                           & LDAER          ;
4249    0030E   /                           & DREAD          ; START A READ IN DATA2 MAP
```

```
LINE    ADDR    STATEMENT

4251            ;*************************************************************************
4252            ;*                                                                       *
4253            ;*        XC.1  or  XC.2                                                 *
4254            ;*        DEF  ADDRESS                                                   *
4255            ;*                                                                       *
4256            ;*        CROSS COMPARE A/B WITH LOCATION IN                             *
4257            ;*        ALTERNATE MAP.  ADDRESS POINTS TO                              *
4258            ;*        A LOCATION IN DATA1 MAP.                                       *
4259            ;*                                                                       *
4260            ;*************************************************************************
4261            ;
4262            XC.10:    CCALL INDMS       & AM2901 TAB,,QREG,PASS,DZ
4263            /                           & CONDEXT MDIR15 ;
4264   0030F    /                           & LDMAR          ;
4265                      CONT              & AM2901 MAPD1,,NOP,PASS,DZ
4266            /                           & LDAER          ;
4267   00310    /                           & DREAD          ; READ FROM DATA1 MAP
4268            XCCONT:   CONT              & AM2901 MAPX,,NOP,PASS,DZ
4269   00311    /                           & LDAER          ; PUT EXEC MAP BACK
4270                      CONT              & AM2901 CAB,,NOP,EXOR,DA
4271   00312    /                           & LODUSR         ; COMPARE DATA
4272                      JP     INCPC      & AM2901 PC,PC,RAMF,ADD,ZA
4273            /                           & CARRYEXT       ;
4274            /                           & CONDUSR NZ     ;
4275            /                           & LDMAR          ; FETCH FROM THE APPROPRIAT
4276   00313    /                           & IFETCH         ; ADDRESS (SKIP IF NO COMPA
4277            XC.20:    CCALL INDMS       & AM2901 TAB,,QREG,PASS,DZ
4278            /                           & CONDEXT MDIR15 ;
4279   00314    /                           & LDMAR          ;
4280                      JP     XCCONT     & AM2901 MAPD2,,NOP,PASS,DZ
4281            /                           & LDAER          ;
4282   00315    /                           & DREAD          ; READ FROM DATA2 MAP
```

```
LINE    ADDR    STATEMENT

4284            ;*********************************************************************
4285            ;*                                                                   *
4286            ;*        XS.1 or  XS.2                                              *
4287            ;*        DEF  ADDRESS                                               *
4288            ;*                                                                   *
4289            ;*        CROSS STORE A/B THROUGH ALTERNATE MAP.                     *
4290            ;*        INDIRECTS ARE RESOLVED IN THE EXECUTE MAP.                 *
4291            ;*        A/B ADDRESSABILITY AND BOOT MODE ARE TURNED                *
4292            ;*        OFF FOR THE FINAL REFERENCE (STORE)                        *
4293            ;*                                                                   *
4294            ;*********************************************************************
4295            ;
4296            XS.10:    CCALL INDMS       & AM2901 TAB,,QREG,PASS,DZ
4297            /                           & CONDEXT MDIR15 ;
4298    00316   /                           & LDMAR          ;
4299                      CONT              & AM2901 MAPD1,,NOP,PASS,DZ
4300    00317   /                           & LDAER          ; LOAD AER WITH DATA1
4301            XSCONT:   CONT              & AM2901 ,CAB,NOP,PASS,ZB
4302            /                           & LDMDOR         ;
4303    00318   /                           & DWRITE         ; WRITE DATA WITH DATA1
4304                      JP    SKIP        & AM2901 MAPX,,NOP,PASS,DZ
4305    00319   /                           & LDAER          ; PUT EXEC MAP BACK
4306            ;
4307            XS.20:    CCALL INDMS       & AM2901 TAB,,QREG,PASS,DZ
4308            /                           & CONDEXT MDIR15 ;
4309    0031A   /                           & LDMAR          ;
4310                      JP    XSCONT      & AM2901 MAPD2,,NOP,PASS,DZ
4311    0031B   /                           & LDAER          ; LOAD AER WITH DATA2
```

```
LINE    ADDR    STATEMENT

4313            ;*************************************************************************
4314            ;*                                                                       *
4315            ;*        LWD1                                                           *
4316            ;*        DEF NEWDATA1                                                   *
4317            ;*                                                                       *
4318            ;*        LOADS THE DATA1 MAP PORTION OF MAPD REG                        *
4319            ;*        FROM NEWDATA1.  THE LOWER 8 BITS OF                            *
4320            ;*        THE DATA WORD ARE LOADED INTO MAPD.                            *
4321            ;*                                                                       *
4322            ;*************************************************************************
4323            ;
4324            LWD1:     CJP   GENMPV     & AM2901 BR,TABQ,QREG,ADD,DZ
4325            /                          & CARRYL          ;
4326            /                          & CONDEXT MPEN    ;
4327            /                          & LUSRCOND        ; LOAD USR WITH INDIRECT BI
4328            /                          & LDMAR           ;
4329   0031C    /                          & CNMGO & DREAD   ; READ NEW DATA1 MAP # OR M
4330                      CCALL MRGIND     & AM2901 ,PC,NOP,PASS,ZB
4331            /                          & CONDUSR SGN     ; IF BIT SET,
4332   0031D    /                          & LDMAR           ; RESOLVE INDIRECTS
4333                      JP    INCPC      & AM2901 TAB,MAPD1,NOP,PASS,DZ
4334   0031E    /                          & IFETCH          ; LOAD NEW DATA1 VALUE, FET
4335            ;
4336            LWD2:     CJP   GENMPV     & AM2901 BR,TABQ,QREG,ADD,DZ
4337            /                          & CARRYL          ;
4338            /                          & CONDEXT MPEN    ; IF MP SYSTEM ON, INTERRUP
4339            /                          & LUSRCOND        ; LOAD USR WITH INDIRECT BI
4340            /                          & LDMAR           ;
4341   0031F    /                          & CNMGO & DREAD   ; READ NEW DATA2 MAP #, ELS
4342                      CCALL MRGIND     & AM2901 ,PC,NOP,PASS,ZB
4343            /                          & CONDUSR SGN     ;
4344   00320    /                          & LDMAR           ; RESOLVE INDIRECTS
4345                      JP    INCPC      & AM2901 TAB,MAPD2,NOP,PASS,DZ
4346   00321    /                          & IFETCH          ; LOAD NEW DATA2 VALUE, FET
```

```
LINE    ADDR    STATEMENT

4348            ;*********************************************************************
4349            ;*                                                                   *
4350            ;*        LPMR                                                       *
4351            ;*                                                                   *
4352            ;*        LOAD PAGE MAPPING REG ADDRESSED BY A REG                   *
4353            ;*        FROM B REG.  A REG IS INCREMENTED                          *
4354            ;*                                                                   *
4355            ;*********************************************************************
4356            ;
4357            LPMR0:    CJP   GENMPV     & AM2901 ,,NOP,ADD,AQ
4358    00322   /                          & CONDEXT MPEN    ; IF MEM PROTECT ENABLED
4359                      CALL  POSIMAPR   & AM2901 ,A,RAMF,ADD,ZB
4360    00323   /                          & CARRYH          ; POSITION MAP REG NUMBER,
4361            ;                                            ; INCREMENT A
4362                      JP    DMSEXIT    & AM2901 ,B,NOP,PASS,ZB
4363    00324   /                          & SPWR MAPWR      ; LOAD MAP REG
4364            ;                                            ; FIX MAR AND FETCH
4365            ;*********************************************************************
4366            ;*                                                                   *
4367            ;*        SPMR                                                       *
4368            ;*                                                                   *
4369            ;*        STORE PAGE MAPPING REG ADDRESSED BY A REG                  *
4370            ;*        INTO B REG.  A REG IS INCREMENTED.                         *
4371            ;*                                                                   *
4372            ;*********************************************************************
4373            ;
4374            SPMR0:    CJP   GENMPV     & AM2901 ,,NOP,ADD,AQ
4375    00325   /                          & CONDEXT MPEN    ; IF MP SYSTEM ON, MP INT
4376                      CALL  POSIMAPR   & AM2901 ,A,RAMF,ADD,ZB
4377    00326   /                          & CARRYH          ; POSITION MAP REG NUMBER;
4378            ;                                            ; INCREMENT A
4379                      JP    DMSEXIT    & AM2901 ,B,RAMF,PASS,DZ
4380    00327   /                          & SPRD PDIRRD     ; B := MAP REG NUMBER
```

```
LINE    ADDR    STATEMENT

4382            ;*********************************************************************
4383            ;*      POSIMAPR - POSITION MAP REG NUMBER                           *
4384            ;*                                                                   *
4385            ;*      This routine loads the AER and the MAR with the              *
4386            ;*      appropriate map register number, and reads the map           *
4387            ;*      register.  The MAR and AER remain set so that the map        *
4388            ;*      register may be written if desired.                          *
4389            ;*                                                                   *
4390            ;*********************************************************************
4391            ;
4392            POSIMAPR: PUSH  H#004       & AM2901 A,S0,RAMF,SUBR,ZA
4393   00328    /                           & CARRYH          ; S0 := A - 1 SR 1
4394                      RFCT              & AM2901 ,S0,SRAMR,PASS,ZB
4395   00329    /                           & SHIFT ROTATE    ; ROTATE S0 RIGHT 4 MORE BI
4396            ;                                       THIS POSITIONS MAPSET # IN LOW 5
4397                      CONT              & AM2901 ,S0,SRAMR,PASS,ZB
4398            /                           & SHIFT ROTATE    ; POSITION PAGE #
4399   0032A    /                           & LDAER           ; LOAD AER WITH MAPSET
4400                      RET               & AM2901 ,S0,NOP,PASS,ZB
4401            /                           & LDMAR           ;
4402   0032B    /                           & SPETC LDMAPD    ; MAR := PAGE #, READ MAP R
```

```
LINE    ADDR    STATEMENT

4404            ;*********************************************************************
4405            ;*                                                                   *
4406            ;*        LDMP                                                       *
4407            ;*        DEF MAP_NUMBER                                             *
4408            ;*        DEF MAP_IMAGE                                              *
4409            ;*                                                                   *
4410            ;*        LOADS THE 32 PAGE REGISTERS IN MAP                         *
4411            ;*        MAP_NUMBER FROM CONSECUTIVE MEMORY                         *
4412            ;*        LOCATIONS STARTING AT MAP_IMAGE                            *
4413            ;*                                                                   *
4414            ;*                                                                   *
4415            ;*        All reads and writes are done in the                       *
4416            ;*        EXECUTE map                                                *
4417            ;*                                                                   *
4418            ;*********************************************************************
4419            ;
4420            LDMP:     CJP   GENMPV     & AM2901 BR,TABQ,QREG,ADD,DZ
4421            /                          & CARRYL          ;
4422            /                          & CONDEXT MPEN    ;
4423            /                          & LUSRCOND        ;
4424            /                          & LDMAR           ;
4425    0032C   /                          & CNMGO & DREAD   ;
4426                      CCALL MRGIND     & AM2901 PC,S7,RAMA,AND,ZQ
4427            /                          & CONDUSR SGN     ;
4428    0032D   /                          & LDMAR           ; RESOLVE MAP_NUMBER
4429                      CALL  READRES    & AM2901 TAB,S0,RAMF,PASS,DZ
4430    0032E   /                          & CREAD           ; RESOLVE MAP_IMAGE;
4431            ;                                            ; START READ OF FIRST DATUM
4432                      LDCT  H#01F      & AM2901 S7,S0,NOP,PASS,ZB
4433    0032F   /                          & LDAER           ; LOAD AER WITH MAP_NUMBER;
4434            ;                                            ; 2901 := # OF PAGES IN A M
4435            ;
4436            ;         LOOP TO TRANSFER MAP
4437            ;
4438            ;  - S7 IS PAGE NUMBER
4439            ;
4440            LDMP5:    CONT             & AM2901 S7,S7,RAMA,ADD,DA
4441            /                          & CARRYL          ;
4442            /                          & LDMAR           ;
4443    00330   /                          & IMM H#0400      ; LOAD MAR WITH PAGE_NUMBER
4444                      CONT             & AM2901 ,,NOP,PASS,DZ
4445    00331   /                          & SPWR MAPWR      ; WRITE THE MAP
4446                      RPCT  LDMP7      & AM2901 MAPX,,NOP,PASS,DZ
4447    00332   /                          & LDAER           ; LOAD AER WITH EXECUTE MAP
4448            TWIEXIT:  JP    INCPC      & AM2901 PC,PC,RAMF,ADD,ZB
4449            /                          & CARRYH          ;
4450            /                          & LDMAR           ;
4451    00333   /                          & IFETCH          ; FETCH NEXT INSTRUCTION
4452            LDMP7:    CONT             & AM2901 ,,QREG,ADD,ZQ
4453            /                          & CARRYH          ;
4454            /                          & LDMAR           ;
4455    00334   /                          & DREAD           ; READ NEXT MAP REGISTER VA
4456                      JP    LDMP5      & AM2901 S7,S0,NOP,PASS,ZB
```

```
LINE    ADDR    STATEMENT

4457    00335   /                       & LDAER         ; LOAD AER WITH MAP_NUMBER;
4458            ;
4459            ;*********************************************************************
4460            ;*                                                                   *
4461            ;*      STMP                                                         *
4462            ;*      DEF MAP_NUMBER                                               *
4463            ;*      DEF MAP_IMAGE                                                *
4464            ;*                                                                   *
4465            ;*      STORES THE 32 PAGE REGISTERS IN MAP                          *
4466            ;*      MAP_NUMBER INTO CONSECUTIVE MEMORY                           *
4467            ;*      LOCATIONS STARTING AT MAP_IMAGE                              *
4468            ;*                                                                   *
4469            ;*      ALL READS AND WRITES ARE DONE IN THE                         *
4470            ;*      EXECUTE MAP                                                  *
4471            ;*********************************************************************
4472            ;
4473            STMP:   CJP   GENMPV    & AM2901 BR,TABQ,QREG,ADD,DZ
4474            /                       & CARRYL        ;
4475            /                       & CONDEXT MPEN  ;
4476            /                       & LUSRCOND      ;
4477            /                       & LDMAR         ;
4478    00336   /                       & CNMGO & DREAD ;
4479                    CCALL MRGIND    & AM2901 PC,S7,RAMA,AND,ZQ
4480            /                       & CONDUSR SGN   ;
4481    00337   /                       & LDMAR         ; RESOLVE MAP_NUMBER
4482            ;
4483                    CALL  INDRES    & AM2901 TAB,S0,RAMF,PASS,DZ
4484    00338   /                       & CREAD         ; RESOLVE MAP_IMAGE
4485            ;
4486            ;       PLACE MAP_IMAGE ADDRESS IN S5
4487            ;
4488                    LDCT  H#01F     & AM2901 S0,S5,RAMA,PASS,ZQ
4489    00339   /                       & LDAER         ; S5 := MAP_IMAGE
4490            ;                                       ; AER := MAP_NUMBER
4491            ;
4492            ;       LOOP TO TRANSFER MAP
4493            ;
4494            ;   - S7 IS PAGE NUMBER REG
4495            ;
4496            STMP5:  CONT            & AM2901 S7,S7,RAMA,ADD,DA
4497            /                       & CARRYL        ;
4498            /                       & IMM H#0400    ;
4499            /                       & LDMAR         ; LOAD MAR WITH PAGE # AND
4500    0033A   /                       & SPETC         ; DO MAP READ (LDMAPD=0)
4501                    CONT            & AM2901 ,,NOP,PASS,DZ
4502            /                       & LDMDOR        ;
4503    0033B   /                       & SPRD PDIRRD   ; PASS MAP_DATA_IN REG TO M
4504                    CONT            & AM2901 S5,S5,RAMA,ADD,ZA
4505            /                       & CARRYH        ;
4506    0033C   /                       & LDMAR         ; LOAD MAR AND INC MAP_IMAG
4507                    RPCT  STMP7     & AM2901 MAPX,,NOP,PASS,DZ
4508            /                       & LDAER         ;
4509    0033D   /                       & DWRITE        ; LOAD AER AND WRITE
4510            ;
```

```
LINE    ADDR    STATEMENT

4511            ;           FETCH NEXT INSTRUCTION
4512            ;
4513                        JP    INCPC       & AM2901 PC,PC,RAMF,ADD,ZB
4514            /                             & CARRYH          ;
4515            /                             & LDMAR           ;
4516    0033E   /                             & IFETCH          ;
4517            STMP7:      JP    STMP5       & AM2901 ,S0,NOP,PASS,ZB
4518    0033F   /                             & LDAER           ; AER = MAP_NUMBER
```

```
LINE    ADDR    STATEMENT

4520            ;*********************************************************************
4521            ;*                                                                   *
4522            ;*        SWMP                                                       *
4523            ;*        DEF   SAVEWMAP                                             *
4524            ;*                                                                   *
4525            ;*        STORES THE CURRENT WMAP INTO MEMORY.                       *
4526            ;*                                                                   *
4527            ;*********************************************************************
4528            SWMP0:    CJP   GENMPV     & AM2901 ,,NOP,ADD,AQ
4529   00340    /                          & CONDEXT MPEN   ; INT IF M.P. SYS ON
4530                      CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
4531            /                          & CARRYL         ;
4532            /                          & CONDEXT MDIR15 ;
4533            /                          & LDMAR          ;
4534   00341    /                          & CMGO & DREAD   ; RESOLVE SAVEWMAP ADDR
4535                      CALL  STWMAP     & AM2901 ,,NOP,ADD,AQ
4536   00342    /                          & SPNOP          ; STORE WMAP
4537                      JZ               & AM2901 PC,PC,RAMA,ADD,ZB
4538            /                          & CARRYH         ;
4539            /                          & LDMAR          ;
4540   00343    /                          & IFETCH         ; FETCH NEXT INSTRUCTION
4541            ;
4542            ;*********************************************************************
4543            ;*                                                                   *
4544            ;*        SIMP                                                       *
4545            ;*        DEF   SAVEIMAP                                             *
4546            ;*                                                                   *
4547            ;*        STORES THE CONTENTS OF IMAP REG INTO                       *
4548            ;*        MEMORY.  IMAP REG IS ACTUALLY IN BOOT                      *
4549            ;*        MEMORY AT LOCATION "IMAPLOC".                              *
4550            ;*                                                                   *
4551            ;*********************************************************************
4552            SIMP0:    CJP   GENMPV     & AM2901 ,,NOP,ADD,AQ
4553   00344    /                          & CONDEXT MPEN   ; INT IF M.P. SYS ON
4554                      CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
4555            /                          & CARRYL         ;
4556            /                          & CONDEXT MDIR15 ;
4557            /                          & LDMAR          ;
4558   00345    /                          & CMGO & DREAD   ; RESOLVE SAVEIMAP ADDR
4559                      CONT             & AM2901 ,,NOP,PASS,DZ
4560            /                          & LDAER          ;
4561   00346    /                          & IMM H#0020     ; ENABLE BOOT MEMORY
4562                      CONT             & AM2901 ,,NOP,PASS,DZ
4563            /                          & IMM IMAPLOC    ;
4564            /                          & LDMAR          ;
4565   00347    /                          & DREAD          ; READ IMAP
4566                      CONT             & AM2901 MAPX,,NOP,PASS,DZ
4567   00348    /                          & LDAER          ; PUT MAPX BACK INTO AER
4568                      CONT             & AM2901 ,,NOP,PASS,ZQ
4569   00349    /                          & LDMAR          ; PUT SAVEIMAP ADDR INTO MA
4570                      JP    SKIP       & AM2901 ,TAB,NOP,PASS,DZ
4571            /                          & LDMDOR         ;
4572   0034A    /                          & DWRITE         ; STORE IMAP INTO MEM
```

```
LINE    ADDR    STATEMENT

4574            ;*********************************************************************
4575            ;*                                                                   *
4576            ;*        XJMP                                                       *
4577            ;*        DEF   NEWWMAP                                              *
4578            ;*        DEF   NEXTINST                                             *
4579            ;*                                                                   *
4580            ;*        XJCQ                                                       *
4581            ;*        DEF   NEWWMAP                                              *
4582            ;*        DEF   NEXTINST                                             *
4583            ;*        DEF   NEWCQ                                                *
4584            ;*                                                                   *
4585            ;*                                                                   *
4586            ;*        LOADS NEW WORKING MAP SET FROM NEWWMAP.                    *
4587            ;*        SETS PC TO NEXTINST AND CONTINUES.                         *
4588            ;*        TURNS MEMORY PROTECT ON IF SIGN BIT OF NEWMAP SET          *
4589            ;*                                                                   *
4590            ;*********************************************************************
4591            ;
4592            XJCQ:     CONT            & AM2901 ,,NOP,ADD,AQ
4593   0034B    /                         & LODMSR SET      ;
4594            XJMP:     CJP   GENMPV    & AM2901 BR,TABQ,QREG,ADD,DZ
4595            /                         & CARRYL          ;
4596            /                         & CONDEXT MPEN    ;
4597            /                         & LUSRCOND        ;
4598            /                         & LDMAR           ;
4599   0034C    /                         & CNMGO & DREAD   ;
4600                      CCALL MRGIND    & AM2901 ,PC,NOP,PASS,ZB
4601            /                         & CONDUSR SGN     ;
4602   0034D    /                         & LDMAR           ; RESOLVE NEWMAP
4603                      CJP   GETCQ     & AM2901 ,,NOP,ADD,AQ
4604            /                         & CONDMSR SGN     ;
4605   0034E    /                         & CREAD           ; GETCQ IF INST WAS XJCQ
4606                      CALL  INDRES    & AM2901 TAB,S1,RAMF,PASS,DZ
4607   0034F    /                         & LODMSR          ; S1 := NEWMAP; RESOLVE NEX
4608            ;
4609            ;     - LOAD MAPX AND MAPD1 AND MAPD2 FROM THE NEW WMAP VALUE
4610            ;
4611            XJ05:     CONT            & AM2901 S1,MAPX,NOP,AND,DA
4612            /                         & LDAER           ;
4613   00350    /                         & IMM H#001F      ; SET EXECUTE MAP
4614                      PUSH  H#001     & AM2901 S1,S1,RAMA,PASS,DZ
4615   00351    /                         & SPRD RL4        ;
4616                      RFCT            & AM2901 ,S1,SRAML,PASS,ZB
4617   00352    /                         & SHIFT ROTATE    ;  POSITION DATA2
4618                      CONT            & AM2901 S1,MAPD2,NOP,AND,DA
4619   00353    /                         & IMM H#001F      ; LOAD DATA2 INTO MAPD2
4620                      CONT            & AM2901 ,S1,SRAML,PASS,ZB
4621            /                         & SHIFT ROTATE    ;
4622   00354    /                         & ENCN SETTDI     ; SET TDI
4623                      CALL  SETAB     & AM2901 S1,S1,RAMA,PASS,DZ
4624   00355    /                         & SPRD RL4        ; POSITION DATA1
4625                      CONT            & AM2901 S1,MAPD1,NOP,AND,DA
4626   00356    /                         & IMM H#001F      ; LOAD DATA1 INTO MAPD1
```

```
LINE    ADDR    STATEMENT

4627            ;
4628            ;       TURN ON MEM PROTECT IF SIGN OF WMAP WAS SET
4629            ;
4630                    CJP   INCFTCH    & AM2901 ,,NOP,PASS,ZQ
4631            /                        & CONDMSR NSGN    ;
4632    00357   /                        & LDMAR           ; MAR := NEXTINST
4633                    JP    INCFTCH    & AM2901 ,,NOP,ADD,AQ
4634    00358   /                        & ENCN SETMPEN    ;  TURN M.P. SYS ON
4635            ;
4636            GETCQ:  CALL  INDRES     & AM2901 TAB,S1,RAMF,PASS,DZ
4637    00359   /                        & LODMSR          ; S1=NEWMAP; MSR = MPEN
4638                    CONT             & AM2901 ,PC,NOP,ADD,ZB
4639            /                        & CARRYH          ;
4640            /                        & LDMAR           ;
4641    0035A   /                        & CREAD           ; READ NEWCQ
4642                    CALL  READRES    & AM2901 ,S2,RAMF,PASS,ZQ
4643    0035B   /                        & SPNOP           ; RESOLVE NEWCQ
4644                    LDCT  XJ05       & AM2901 ,S2,QREG,PASS,ZB
4645    0035C   /                        & SPETC SETCS     ; QREG = NEXTINST; TURN CSM
4646                    JRP   $+1        & AM2901 TAB,Q,RAMF,PASS,DZ
4647    0035D   /                        & CONDEXT SIGN    ;  IF CSMODE S.B. ON, JR
4648                    JRP              & AM2901 Q,Q,SRAMR,ADD,AB
4649            /                        & CARRYL          ;
4650            /                        & CONDEXT         ; CONDEXT IS MPEN (ALWAYS J
4651            /                        & SHIFT B#0000    ; Q<15> = 0
4652    0035E   /                        & SPETC CLRCS     ; TURN OFF CSMODE
```

```
LINE    ADDR    STATEMENT

4654            ;*********************************************************************
4655            ;*                                                                   *
4656            ;*       DMS Move Instruction Group                                  *
4657            ;*                                                                   *
4658            ;*  Move Words:                                                      *
4659            ;*                                                                   *
4660            ;*       MW00 - move words from EXECUTE to EXECUTE                   *
4661            ;*       MW01 - move words from EXECUTE to DATA1                     *
4662            ;*       MW02 - move words from EXECUTE to DATA2                     *
4663            ;*       MW10 - move words from DATA1 to EXECUTE                     *
4664            ;*       MW11 - move words from DATA1 to DATA1                       *
4665            ;*       MW12 - move words from DATA1 to DATA2                       *
4666            ;*       MW20 - move words from DATA2 to EXECUTE                     *
4667            ;*       MW21 - move words from DATA2 to DATA1                       *
4668            ;*       MW22 - move words from DATA2 to DATA2                       *
4669            ;*                                                                   *
4670            ;*  Move Bytes:                                                      *
4671            ;*                                                                   *
4672            ;*       MB00 - move bytes from EXECUTE to EXECUTE                   *
4673            ;*       MB01 - move bytes from EXECUTE to DATA1                     *
4674            ;*       MB02 - move bytes from EXECUTE to DATA2                     *
4675            ;*       MB10 - move bytes from DATA1 to EXECUTE                     *
4676            ;*       MB11 - move bytes from DATA1 to DATA1                       *
4677            ;*       MB12 - move bytes from DATA1 to DATA2                       *
4678            ;*       MB20 - move bytes from DATA2 to EXECUTE                     *
4679            ;*       MB21 - move bytes from DATA2 to DATA1                       *
4680            ;*       MB22 - move bytes from DATA2 to DATA2                       *
4681            ;*                                                                   *
4682            ;*  Operation:                                                       *
4683            ;*                                                                   *
4684            ;*       The A register contains the source address.  The            *
4685            ;*       B register contains the destination address. The            *
4686            ;*       X register contains the word count which can be             *
4687            ;*       zero.  When the instruction completes, A and B              *
4688            ;*       are incremented by the number of words moved and            *
4689            ;*       X is zero.  These instructions are interruptible.           *
4690            ;*                                                                   *
4691            ;*********************************************************************
```

```
LINE    ADDR    STATEMENT

4693            ;*****
4694            ;*****    MW00/MB00 - EXEC TO EXEC
4695            ;*****
4696            MOV00:    LDCT  MOVBYTAB   & AM2901 MAPX,S6,RAMF,PASS,DZ
4697            /                          & LODMSR SET      ; SAME MAP
4698   0035F    /                          & LDAER           ; AER = FROM
4699                      JRP   MOVWRDAB   & AM2901 MAPX,S7,RAMF,PASS,DZ
4700   00360    /                          & CONDEXT IR11    ;
4701            ;
4702            ;         MW01/MB01 - EXEC TO DATA1
4703            ;
4704            MOV01:    LDCT  MOVBYTA    & AM2901 MAPX,S6,RAMF,PASS,DZ
4705            /                          & LODMSR RESET    ; DIFFERENT MAP
4706   00361    /                          & LDAER           ; AER = FROM
4707                      JRP   MOVWRDA    & AM2901 MAPD1,S7,RAMF,PASS,DZ
4708   00362    /                          & CONDEXT IR11    ;
4709            ;
4710            ;         MW02/MB02 - EXEC TO DATA2
4711            ;
4712            MOV02:    LDCT  MOVBYTA    & AM2901 MAPX,S6,RAMF,PASS,DZ
4713            /                          & LODMSR RESET    ; DIFFERENT MAP
4714   00363    /                          & LDAER           ; AER = FROM
4715                      JRP   MOVWRDA    & AM2901 MAPD2,S7,RAMF,PASS,DZ
4716   00364    /                          & CONDEXT IR11    ;
4717            ;
4718            ;         MW10/MB10 - DATA1 TO EXEC
4719            ;
4720            MOV10:    LDCT  MOVBYTB    & AM2901 MAPD1,S6,RAMF,PASS,DZ
4721            /                          & LODMSR RESET    ; DIFFERENT MAP
4722   00365    /                          & LDAER           ; AER = FROM
4723                      JRP   MOVWRDB    & AM2901 MAPX,S7,RAMF,PASS,DZ
4724   00366    /                          & CONDEXT IR11    ;
4725            ;
4726            ;         MW11/MB11 - DATA1 TO DATA1
4727            ;
4728            MOV11:    LDCT  MOVEBYTE   & AM2901 MAPD1,S6,RAMF,PASS,DZ
4729            /                          & LODMSR SET      ; SAME MAP
4730   00367    /                          & LDAER           ; AER = FROM
4731                      JRP   MOVWORD    & AM2901 MAPD1,S7,RAMF,PASS,DZ
4732   00368    /                          & CONDEXT IR11    ;
4733            ;
4734            ;         MW12/MB12 - DATA1 TO DATA2
4735            ;
4736            MOV12:    LDCT  MOVEBYTE   & AM2901 MAPD1,S6,RAMF,PASS,DZ
4737            /                          & LODMSR RESET    ; DIFFERENT MAP
4738   00369    /                          & LDAER           ; AER = FROM
4739                      JRP   MOVWORD    & AM2901 MAPD2,S7,RAMF,PASS,DZ
4740   0036A    /                          & CONDEXT IR11    ;
4741            ;
4742            ;         MW20/MB20 - DATA2 TO EXEC
4743            ;
4744            MOV20:    LDCT  MOVBYTB    & AM2901 MAPD2,S6,RAMF,PASS,DZ
4745            /                          & LODMSR RESET    ; DIFFERENT MAP
```

```
LINE    ADDR    STATEMENT

4746    0036B   /                       & LDAER          ; AER = FROM
4747                    JRP   MOVWRDB   & AM2901 MAPX,S7,RAMF,PASS,DZ
4748    0036C   /                       & CONDEXT IR11   ;
4749            ;
4750            ;       MW21/MB21 - DATA2 TO DATA1
4751            ;
4752            MOV21:  LDCT  MOVEBYTE  & AM2901 MAPD2,S6,RAMF,PASS,DZ
4753            /                       & LODMSR RESET   ; DIFFERENT MAP
4754    0036D   /                       & LDAER          ; AER = FROM
4755                    JRP   MOVWORD   & AM2901 MAPD1,S7,RAMF,PASS,DZ
4756    0036E   /                       & CONDEXT IR11   ;
4757            ;
4758            ;       MW22/MB22 - DATA2 TO DATA2
4759            ;
4760            MOV22:  LDCT  MOVEBYTE  & AM2901 MAPD2,S6,RAMF,PASS,DZ
4761            /                       & LODMSR SET     ; SAME MAP
4762    0036F   /                       & LDAER          ; AER = FROM
4763                    JRP   MOVWORD   & AM2901 MAPD2,S7,RAMF,PASS,DZ
4764    00370   /                       & CONDEXT IR11   ;
```

```
LINE    ADDR    STATEMENT

4766            ;*********************************************************************
4767            ;*                                                                   *
4768            ;*        MOVWORD                                                    *
4769            ;*        -------                                                    *
4770            ;*                                                                   *
4771            ;*        This routine performs all the cross map move word          *
4772            ;*          instructions.  At entry, S6 is the FROM map and          *
4773            ;*          S7 is the TO map.  If the FROM map and the TO            *
4774            ;*          map are the same, the MSR is set, else reset.            *
4775            ;*                                                                   *
4776            ;*********************************************************************
4777            ;
4778            MOVWRDA:  JP    MOVWORD    & AM2901 BREL,A,RAMF,ADD,ZB
4779   00371    /                          & CARRYL          ; BREL A
4780            MOVWRDAB: CONT             & AM2901 BREL,A,RAMF,ADD,ZB
4781   00372    /                          & CARRYL          ; BREL A, THEN B
4782            MOVWRDB:  CONT             & AM2901 BREL,B,RAMF,ADD,ZB
4783   00373    /                          & CARRYL          ; BREL B
4784            MOVWORD:  LDCT  DMSEXIT    & AM2901 ,X,NOP,PASS,ZB
4785   00374    /                          & LODUSR          ; USR = COUNT STATUS
4786            MVWLOOP:  JRP   MVW1       & AM2901 A,A,RAMA,ADD,ZB
4787            /                          & CARRYEXT        ; INC A IF NOT DONE
4788            /                          & CONDUSR NZ      ; IF ZERO, DONE,
4789            /                          & LDMAR           ;
4790   00375    /                          & CMGO & DREAD    ; ELSE READ WORD
4791            MVW1:     CJP   SAMEMAP    & AM2901 B,B,RAMA,ADD,ZB
4792            /                          & CARRYH          ;
4793            /                          & CONDMSR SGN     ;
4794   00376    /                          & LDMAR           ; MAR := "TO" ADDRESS
4795                      CONT             & AM2901 ,S7,NOP,PASS,ZB
4796   00377    /                          & LDAER           ;
4797            SAMEMAP:  CJP   INTMVW     & AM2901 ,,NOP,PASS,DZ
4798            /                          & CONDEXT INTRPT ; IF INT PEND, INTERRUPT
4799            /                          & LDMDOR          ;
4800   00378    /                          & DWRITE          ;
4801                      JP    MVWLOOP    & AM2901 S6,X,RAMA,SUBR,ZB
4802            /                          & CARRYH          ;
4803            /                          & LODUSR          ;
4804   00379    /                          & LDAER           ; LOOP
```

```
LINE    ADDR    STATEMENT

4806            ;*********************************************************************
4807            ;*                                                                   *
4808            ;*        MOVBYTE                                                    *
4809            ;*        -------                                                    *
4810            ;*                                                                   *
4811            ;*        This routine performs all the cross map move byte          *
4812            ;*          instructions.  At entry, S6 is the FROM map and          *
4813            ;*          S7 is the TO map.  If the FROM map and the TO map        *
4814            ;*          are the same, the MSR is SET, else the MSR is RESET.     *
4815            ;*                                                                   *
4816            ;*********************************************************************
4817            ;
4818            MOVBYTA:  CONT             & AM2901 ,A,SRAMR,PASS,ZB
4819   0037A    /                          & SHIFT ROTATE    ; A = WORD A
4820                      CONT             & AM2901 BREL,A,SRAML,ADD,ZB
4821            /                          & CARRYL          ;
4822   0037B    /                          & SHIFT ROTATE    ; BREL, RESTORE A
4823                      JP     MOVEBYTE  & AM2901 ,,NOP,ADD,AQ
4824   0037C    /                          & SPNOP           ;
4825            MOVBYTAB: CONT             & AM2901 ,A,SRAMR,PASS,ZB
4826   0037D    /                          & SHIFT ROTATE    ; A=WORD A
4827                      CONT             & AM2901 BREL,A,SRAML,ADD,ZB
4828            /                          & CARRYL          ;
4829   0037E    /                          & SHIFT ROTATE    ; BREL A, RESTORE A
4830            MOVBYTB:  CONT             & AM2901 ,B,SRAMR,PASS,ZB
4831   0037F    /                          & SHIFT ROTATE    ; B=WORD B
4832                      CONT             & AM2901 BREL,B,SRAML,ADD,ZB
4833            /                          & CARRYL          ;
4834   00380    /                          & SHIFT ROTATE    ; BREL B
4835            MOVEBYTE: CONT             & AM2901 S3,S3,RAMF,PASS,DZ
4836   00381    /                          & IMM H#FF00      ; S3 = EVEN BYTE MASK
4837                      CONT             & AM2901 S3,S5,RAMF,EXNOR,ZA
4838   00382    /                          & SPNOP           ; S5 = ODD BYTE MASK
4839                      CONT             & AM2901 S6,X,NOP,PASS,ZB
4840   00383    /                          & LODUSR          ; USR = X
4841            LMOVBYT:  CJP    LMOVBYT1  & AM2901 S6,B,RAMA,ADD,ZB
4842            /                          & CARRYEXT        ; INC B IF NOT DONE
4843            /                          & CONDUSR NZ      ;
4844   00384    /                          & LDAER           ; SET FROM MAP
4845            DMSEXIT:  JP     FIXMAR    & AM2901 MAPX,,NOP,PASS,DZ
4846   00385    /                          & LDAER           ; AER = MAPX
4847            ;
4848            LMOVBYT1: CONT             & AM2901 A,S0,SRAMR,PASS,ZA
4849   00386    /                          & SHIFT ROTATE    ; MAKE S0 "FROM" WORD ADDRE
4850                      CONT             & AM2901 B,S1,SRAMR,SUBR,ZA
4851            /                          & CARRYH          ; B IS ONE TOO MANY
4852   00387    /                          & SHIFT ROTATE    ; MAKE S1 "TO" WORD ADDRESS
4853                      CONT             & AM2901 ,S0,NOP,PASS,ZB
4854            /                          & LODUSR          ;
4855            /                          & LDMAR           ;
4856   00388    /                          & DREAD           ; READ "FROM" WORD
4857                      CJP    MFMOD     & AM2901 S7,S1,NOP,PASS,ZB
4858            /                          & CONDUSR SGN     ; IF SIGN, FROM ODD
```

```
LINE    ADDR    STATEMENT

4859    00389   /                       & LDMAR           ; MAR = "TO" ADDRESS
4860                    CJP   MEVTOEV   & AM2901 S7,A,RAMA,ADD,ZB
4861            /                       & CARRYH          ; INC A
4862            /                       & CONDUSR NSGN    ; IF SIGN, TO ODD
4863            /                       & LDAER           ; AER = "TO" MAP
4864    0038A   /                       & DREAD           ; READ "TO" WORD
4865                    CALL  BYTESWAP  & AM2901 S3,S4,RAMF,AND,DA
4866    0038B   /                       & SPNOP           ; S4 = SWAPPED "FROM" BYTE
4867            MEVSAVE:  CONT          & AM2901 S3,S2,RAMF,AND,DA
4868    0038C   /                       & SPNOP           ; S2 = "TO" EVEN BYTE
4869            MWRBYTE:  LDCT  LMOVBYT & AM2901 S2,S4,NOP,OR,AB
4870            /                       & LDMDOR          ;
4871    0038D   /                       & DWRITE          ; WRITE THE "TO" WORD
4872            INTMVW:   JRP   INTERRPT & AM2901 S6,X,RAMF,SUBR,ZB
4873            /                       & CARRYH          ; DEC COUNT
4874            /                       & CONDEXT INTRPT ; IF INT PENDING, SERVICE I
4875    0038E   /                       & LUSRCOND        ; LOAD USR WITH COUNT STATU
4876            ;
4877            MEVTOEV:  JP    MODSAVE & AM2901 S3,S4,RAMF,AND,DA
4878    0038F   /                       & SPNOP           ; S4 = FROM BYTE
4879            MFMOD:    CJP   MODTOOD & AM2901 S7,A,RAMA,ADD,ZB
4880            /                       & CARRYH          ; INC A
4881            /                       & CONDUSR SGN     ; IF "TO" IS ODD
4882            /                       & LDAER           ; AER = "TO" MAP
4883    00390   /                       & DREAD           ; READ "TO" WORD
4884            MODTOEV:  CALL  BYTESWAP & AM2901 S5,S4,RAMF,AND,DA
4885    00391   /                       & SPNOP           ; S4 = FROM BYTE, SWAP IT
4886            MODSAVE:  JP    MWRBYTE & AM2901 S5,S2,RAMF,AND,DA
4887    00392   /                       & SPNOP           ; S2 = ODD "TO" BYTE
4888            ;
4889            MODTOOD:  JP    MEVSAVE & AM2901 S5,S4,RAMF,AND,DA
4890    00393   /                       & SPNOP           ; S4 = FROM BYTE
```

```
LINE    ADDR    STATEMENT

4892            ;*********************************************************************
4893            ;*                                                                   *
4894            ;*        INDMS - RESOLVE INDIRECT WITHOUT MAKING THE FINAL          *
4895            ;*                ADDRESS BASE RELATIVE, AND WITHOUT READ            *
4896            ;*                OF OPERAND                                         *
4897            ;*                ( USED BY DMS CROSS MAP INSTRUCTIONS)              *
4898            ;*                                                                   *
4899            ;*********************************************************************
4900            ;
4901            INDMS:    LDCT   H#001     & AM2901 BR,TABQ,QREG,ADD,DZ
4902            /                          & CARRYL         ;
4903            /                          & LDMAR          ;
4904   00394    /                          & DREAD          ;
4905            INDMS1:   CRET             & AM2901 TAB,,QREG,PASS,DZ
4906            /                          & CONDEXT        ;
4907            /                          & MWAIT          ; CONDEXT IS NSIGN
4908   00395    /                          & LDMAR          ;
4909                      RPCT   INDMS1    & AM2901 BR,TABQ,QREG,ADD,DZ
4910            /                          & CARRYL         ;
4911            /                          & LDMAR          ;
4912   00396    /                          & DREAD          ;
4913                      CRET             & AM2901 TAB,,QREG,PASS,DZ
4914            /                          & CONDEXT        ;
4915            /                          & MWAIT          ; CONDEXT IS NSIGN
4916   00397    /                          & LDMAR          ;
4917                      CJP    INTERRPT  & AM2901 ,,NOP,ADD,AQ
4918   00398    /                          & CONDEXT INTRPT ;
4919                      JP     INDMS     & AM2901 ,,NOP,ADD,AQ
4920   00399    /                          & SPNOP          ;
```

```
LINE    ADDR    STATEMENT

4922            ;
4923            ;       MBYTE - MOVE BYTES A TO B, S0 TIMES >0
4924            ;
4925            MBYTE:  CONT            & AM2901 ,A,SRAMR,PASS,ZB
4926    0039A   /                       & SHIFT ROTATE   ; A = SOURCE WORD ADDRESS
4927                    CONT            & AM2901 ,B,SRAMR,PASS,ZB
4928    0039B   /                       & SHIFT ROTATE   ; B := DEST WORD ADDRESS
4929                    CONT            & AM2901 ,S0,SRAMQR,PASS,ZB
4930    0039C   /                       & SHIFT B#0110   ; S0 := WORD COUNT
4931                    CONT            & AM2901 ,S0,RAMF,SUBR,ZB
4932            /                       & CARRYH         ;
4933    0039D   /                       & LODMSR         ; S0 := WC-1; MSR=USR=STATU
4934                    CJP   ONEBYTE   & AM2901 ,A,NOP,PASS,ZB
4935            /                       & CONDUSR SGN    ; IF BYTE COUNT WAS 1, JP
4936            /                       & LDMAR          ; USR = A
4937    0039E   /                       & DREAD          ; READ SOURCE WORD
4938                    CJP   FROMOD    & AM2901 ,B,NOP,PASS,ZB
4939            /                       & CONDUSR SGN    ;
4940    0039F   /                       & LDMAR          ; IF SOUCE ADDRESS ODD, JP
4941                    CJP   EVTOOD    & AM2901 TAB,S4,RAMF,PASS,DZ
4942            /                       & CONDUSR SGN    ; IF 'TO' IS ODD, JP AND RE
4943    003A0   /                       & CMGO & DREAD   ; DEST WORD.  S4 = SOURCE W
4944            EVLOOP: CJP   ENDEV     & AM2901 TAB,TAB,NOP,PASS,DZ
4945            /                       & CONDMSR Z      ;
4946            /                       & LDMDOR         ;
4947    003A1   /                       & DWRITE         ; WRITE 2 BYTES; END IF S0=
4948            EVENT:  CONT            & AM2901 ,A,RAMF,ADD,ZB
4949            /                       & CARRYH         ;
4950    003A2   /                       & LDMAR          ; INC A; MAR=NEXT SOURCE AD
4951                    CJP   INTEV     & AM2901 ,B,RAMF,ADD,ZB
4952            /                       & CARRYH         ;
4953            /                       & CONDEXT INTRPT ;
4954    003A3   /                       & CNMGO & DREAD  ; CHECK FOR INT; INC B
4955                    JP    EVLOOP    & AM2901 B,S0,RAMA,SUBR,ZB
4956            /                       & CARRYH         ; DEC S0
4957            /                       & LODMSR         ; MSR = STATUS
4958    003A4   /                       & LDMAR ;        ; MAR=NEXT DEST ADDR
4959            ENDEV:  CONT            & AM2901 ,A,SRAML,ADD,ZB
4960            /                       & CARRYH         ; INC A
4961            /                       & SHIFT ROTATE   ; RESTORE A TO BYTE ADDRESS
4962    003A5   /                       & LDMAR          ; MAR = LAST SRC ADDR
4963                    CONT            & AM2901 B,,NOP,EXOR,AQ
4964    003A6   /                       & LODMSR         ; DO COUNT AND DEST ADDR HA
4965            ;                                        ;   ODD PARITY?
4966                    CONT            & AM2901 ,B,SRAML,ADD,ZB
4967            /                       & CARRYH         ;
4968    003A7   /                       & SHIFT ROTATE   ; INC B ; RESTORE B TO BYTE
4969                    CJP   SKIP      & AM2901 ,,NOP,ADD,AQ
4970    003A8   /                       & CONDMSR NSGN   ; IF EVEN PARITY, DONE
4971                    CONT            & AM2901 B,S1,SRAMR,PASS,ZA
4972            /                       & SHIFT ROTATE   ; S1 = DEST WORD ADDRESS
4973    003A9   /                       & DREAD          ; READ SOURCE WORD
4974                    CONT            & AM2901 S1,A,RAMA,ADD,ZB
```

```
LINE   ADDR   STATEMENT

4975           /                          & CARRYH          ; INC A
4976   003AA   /                          & LDMAR           ; LOAD MAR WITH DEST ADDR
4977                   CONT               & AM2901 TAB,S4,RAMF,PASS,DZ
4978   003AB   /                          & DREAD           ; S4=SRC WORD ; READ DEST W
4979           STEV:   CONT               & AM2901 S4,,QREG,AND,DA
4980   003AC   /                          & IMM H#FF00      ; Q:= SRC BYTE
4981                   CONT               & AM2901 TAB,S3,RAMF,PASS,DZ
4982   003AD   /                          & SPNOP           ; S3=DEST WORD
4983                   CONT               & AM2901 S3,S3,RAMF,NOTRS,DA
4984   003AE   /                          & IMM H#FF00      ; KEEP ODD BYTE
4985                   CONT               & AM2901 S3,TAB,NOP,OR,AQ
4986           /                          & LDMDOR          ;
4987   003AF   /                          & DWRITE          ; OR BYTES; WRITE WORD
4988           INCBPC: JP     INCPC       & AM2901 PC,B,RAMA,ADD,ZB
4989           /                          & CARRYH          ;
4990           /                          & LDMAR           ;
4991   003B0   /                          & IFETCH          ; INC B; FETCH NEXT INST
4992           EVTOOD: CONT               & AM2901 ,S0,SRAMQL,PASS,ZB
4993   003B1   /                          & SHIFT B#0110    ; RESTORE BYTE COUNT TO S0
4994                   CONT               & AM2901 ,S0,SRAMQR,SUBR,ZB
4995           /                          & CARRYH          ; DEC BC
4996           /                          & SHIFT B#0110    ; S0 = WORD COUNT
4997   003B2   /                          & LODMSR          ; MSR=BC
4998                   CONT               & AM2901 A,A,RAMF,OR,DA
4999   003B3   /                          & IMM H#8000      ; INC BA IN A
5000                   CONT               & AM2901 B,B,RAMF,ADD,DA
5001           /                          & CARRYL          ;
5002   003B4   /                          & IMM H#8001      ; INC BA IN B
5003                   CALL   BYTESWAP    & AM2901 TAB,S1,RAMF,PASS,DZ
5004   003B5   /                          & SPNOP           ; S1=INIT DEST WORD; SWAP S
5005                   CONT               & AM2901 S1,S1,RAMF,AND,DA
5006   003B6   /                          & IMM H#FF00      ; S1=EVEN BYTE
5007           ODLOOP: CONT               & AM2901 S4,S3,RAMF,NOTRS,DA
5008   003B7   /                          & IMM H#FF00      ; S3=ODD BYTE
5009                   CJP    ENDOD       & AM2901 S3,S1,NOP,OR,AB
5010           /                          & CONDMSR SGN     ;
5011           /                          & LDMDOR          ;
5012   003B8   /                          & DWRITE          ; WRITE DEST WORD; IF WC<0,
5013           ODENT:  CONT               & AM2901 S4,S1,RAMF,AND,DA
5014   003B9   /                          & IMM H#FF00      ; S1=EVEN BYTE
5015                   CJP    INTOD       & AM2901  ,A,RAMF,ADD,ZB
5016           /                          & CARRYH          ; INC A
5017           /                          & CONDEXT INTRPT  ; CHECK FOR INTERRUPT
5018           /                          & LDMAR           ;
5019   003BA   /                          & CNMGO & DREAD   ;
5020                   CONT               & AM2901 ,S0,RAMF,SUBR,ZB
5021           /                          & CARRYH          ; DEC WORD COUNT
5022   003BB   /                          & LODMSR          ; MSR = WORD COUNT STATUS
5023                   CALL   BYTESWAP    & AM2901 TAB,S4,RAMF,PASS,DZ
5024   003BC   /                          & SPNOP           ; S4=SWAPPED NEXT SRC WORD
5025                   JP     ODLOOP      & AM2901 B,B,RAMA,ADD,ZB
5026           /                          & CARRYH          ; INC B
5027   003BD   /                          & LDMAR           ; MAR=DEST ADDR
5028           ENDOD:  CONT               & AM2901 ,A,SRAML,ADD,ZB
```

```
LINE    ADDR    STATEMENT

5029            /                       & CARRYEXT        ;
5030            /                       & CONDMSR Z       ;
5031    003BE   /                       & SHIFT ROTATE    ; RESTORE A
5032                    LDCT  SKIP      & AM2901 B,,NOP,EXOR,AQ
5033    003BF   /                       & LODMSR          ; MSR = PARITY
5034                    CONT            & AM2901 ,B,SRAML,PASS,ZB
5035            /                       & SHIFT ROTATE    ;
5036    003C0   /                       & LDMAR           ; RESTORE B ; MAR = DEST AD
5037                    JRP   STEV      & AM2901 ,A,RAMF,ADD,ZB
5038            /                       & CARRYEXT        ; IF ODD PARITY: JP,
5039            /                       & CONDMSR SGN     ; STORE BYTE, AND INC A
5040    003C1   /                       & CMGO & DREAD    ; ELSE DONE (JR)
5041            FROMOD: CJP   ODTOOD    & AM2901 TAB,S4,RAMF,PASS,DZ
5042            /                       & CONDUSR SGN     ; IF 'TO' ODD, JP & READ DE
5043    003C2   /                       & CMGO & DREAD    ; R4 = SOURCE WORD
5044                    CONT            & AM2901 S4,S4,RAMA,PASS,DZ
5045    003C3   /                       & SPRD RL4        ; SWAP R4
5046                    JP    ODENT     & AM2901 S4,S4,RAMA,PASS,DZ
5047    003C4   /                       & SPRD RL4        ; AND JP TO ODENT
5048            ODTOOD: CONT            & AM2901 B,B,RAMA,NOTRS,DA
5049    003C5   /                       & IMM H#8000      ; DEC BA IN B
5050                    CONT            & AM2901 TAB,S2,RAMF,PASS,DZ
5051    003C6   /                       & SPNOP           ; S2 = DEST WORD
5052                    CONT            & AM2901 S2,S3,RAMF,AND,DA
5053    003C7   /                       & IMM H#FF00      ; Q = EVEN BYTE
5054                    CONT            & AM2901 S4,S4,RAMF,NOTRS,DA
5055    003C8   /                       & IMM H#FF00      ; S4 = ODD BYTE
5056                    LDCT  EVENT     & AM2901 S4,S3,NOP,OR,AB
5057            /                       & LDMDOR          ;
5058    003C9   /                       & DWRITE          ; WRITE ONE BYTE
5059                    CONT            & AM2901 A,A,RAMF,NOTRS,DA
5060    003CA   /                       & IMM H#8000      ; DEC BA IN A
5061                    CONT            & AM2901 ,S0,SRAMQL,PASS,ZB
5062    003CB   /                       & SHIFT B#0110    ; S0=BYTE COUNT
5063                    CONT            & AM2901 ,S0,SRAMQR,SUBR,ZB
5064            /                       & CARRYH          ;
5065            /                       & SHIFT B#0110    ;
5066    003CC   /                       & LODUSR          ; DEC BC ; S0 = WC ; USR=BC
5067                    JRP   ENDEV     & AM2901 ,S0,RAMF,ADD,ZB
5068            /                       & CARRYH          ;
5069    003CD   /                       & CONDUSR SGN     ; FIX S0 FOR EVENT
5070            ;
5071            ONEBYTE: CONT           & AM2901 B,S1,RAMA,EXNOR,DZ
5072            /                       & IMM H#FF00      ; S1 = BYTE MASK
5073    003CE   /                       & LDMAR           ; MAR = DEST ADDR
5074                    CONT            & AM2901 A,B,NOP,EXOR,AB
5075    003CF   /                       & LODMSR          ; MSR = PARITY
5076                    CONT            & AM2901 ,A,SRAML,PASS,ZB
5077            /                       & LODUSR          ; USR = A
5078            /                       & SHIFT ROTATE    ; RESTORE A
5079    003D0   /                       & DREAD           ; READ DEST
5080                    CJP   FMOD      & AM2901 TAB,S4,RAMF,PASS,DZ
5081    003D1   /                       & CONDUSR SGN     ; S4=SOURCE WORD;IF FROM OD
5082                    CONT            & AM2901 S1,S4,RAMF,NOTRS,AB
```

```
LINE    ADDR    STATEMENT

5083    003D2   /                           & SPNOP           ; S4 = EVEN SOURCE BYTE
5084            ONE.A:      CONT            & AM2901 ,B,SRAML,PASS,ZB
5085            /                           & LODUSR          ; USR = B
5086    003D3   /                           & SHIFT ROTATE    ; RESTORE B
5087                        CJP    TOOD     & AM2901 TAB,S3,RAMF,PASS,DZ
5088    003D4   /                           & CONDUSR SGN     ; S3 = DEST WORD ; IF DEST
5089                        CONT            & AM2901 S1,S3,QREG,AND,AB
5090    003D5   /                           & SPNOP           ; Q = ODD DEST BYTE
5091            ONE.B:      CCALL BYTESWAP  & AM2901 ,A,RAMF,ADD,ZB
5092            /                           & CARRYH          ; INC A
5093    003D6   /                           & CONDMSR SGN     ; IF ODD PARITY, SWAP S4
5094                        JP     INCBPC   & AM2901 S4,TAB,NOP,OR,AQ
5095            /                           & LDMDOR          ;
5096    003D7   /                           & DWRITE          ; WRITE BYTE, GO INC B AND
5097            FMOD:       JP     ONE.A    & AM2901 S1,S4,RAMF,AND,AB
5098    003D8   /                           & SPNOP           ; S4 = ODD SOURCE BYTE
5099            TOOD:       JP     ONE.B    & AM2901 S1,S3,QREG,NOTRS,AB
5100    003D9   /                           & SPNOP           ; Q = EVEN DEST BYTE
5101            ;
5102            INTOD:      CONT            & AM2901 ,A,RAMF,SUBR,ZB
5103    003DA   /                           & CARRYH          ; FIX A
5104                        CONT            & AM2901 ,S0,RAMF,ADD,ZB
5105    003DB   /                           & CARRYH          ; INC S0
5106            INTEV:      CONT            & AM2901 ,A,SRAML,PASS,ZB
5107    003DC   /                           & SHIFT ROTATE    ; RESTORE A
5108                        CONT            & AM2901 ,B,SRAML,PASS,ZB
5109    003DD   /                           & SHIFT ROTATE    ; RESTORE B
5110                        CONT            & AM2901 ,S0,SRAMQL,PASS,ZB
5111    003DE   /                           & SHIFT B#0110    ; RESTORE COUNT
5112                        JP     INTPEND  & AM2901 ,S0,RAMF,SUBR,ZB
5113    003DF   /                           & CARRYH          ; MODIFY S0 FOR INTPEND
```

```
LINE   ADDR   STATEMENT

5115          FILL:     CONT             & AM2901 ,A,SRAMR,PASS,ZB
5116   003E0  /                          & SHIFT ROTATE    ; A = SRC WORD ADDRESS
5117                     CONT             & AM2901 ,B,SRAMR,PASS,ZB
5118   003E1  /                          & SHIFT ROTATE    ; B = DEST WORD ADDRESS
5119                     CONT             & AM2901 ,S0,SRAMQR,PASS,ZB
5120   003E2  /                          & SHIFT B#0110    ; S0 = WORD COUNT
5121                     CONT             & AM2901 A,S0,RAMA,SUBR,ZB
5122          /                          & CARRYH          ;
5123          /                          & LODMSR          ; S0 = MSR = WC - 1
5124          /                          & LDMAR           ;
5125   003E3  /                          & DREAD           ; READ SOURCE WORD
5126                     CJP   ONEBYTE    & AM2901 ,B,NOP,PASS,ZB
5127          /                          & CONDUSR SGN     ; IF WC-1 IS NEG, JP
5128   003E4  /                          & LDMAR           ; MAR = DEST ADDR
5129                     CJP   FILLTOOD   & AM2901 TAB,S3,RAMF,PASS,DZ
5130   003E5  /                          & CONDUSR SGN     ;  S3=SRC WORD;JP&MGO IF DE
5131                     CONT             & AM2901 S3,S3,RAMF,NOTRS,DA
5132   003E6  /                          & IMM H#FF00      ;
5133                     CALL  L4MORE     & AM2901 S3,S4,RAMA,PASS,DZ
5134   003E7  /                          & SPRD RL4        ;
5135          FILLCONT: CONT             & AM2901 S3,S4,RAMF,OR,AB
5136          /                          & LDMDOR          ;
5137   003E8  /                          & DWRITE          ;
5138          FILLOOP:  CJP   ENDOD      & AM2901 ,B,RAMF,ADD,ZB
5139          /                          & CARRYH          ;
5140          /                          & CONDMSR Z       ;
5141   003E9  /                          & LDMAR           ;
5142                     CJP   INTEV      & AM2901 ,A,RAMF,ADD,ZB
5143          /                          & CARRYH          ;
5144   003EA  /                          & CONDEXT INTRPT  ;
5145                     JP    FILLOOP    & AM2901 ,S0,RAMF,SUBR,ZB
5146          /                          & CARRYH          ;
5147          /                          & LODMSR          ;
5148   003EB  /                          & DWRITE          ;
5149          FILLTOOD: CONT             & AM2901 S3,S4,RAMF,AND,DA
5150   003EC  /                          & IMM H#FF00      ;
5151                     CALL  BYTESWAP   & AM2901 TAB,S2,RAMF,PASS,DZ
5152   003ED  /                          & SPNOP           ;
5153                     CONT             & AM2901 S2,S2,RAMF,AND,DA
5154   003EE  /                          & IMM H#FF00      ;
5155                     CONT             & AM2901 S2,S4,NOP,OR,AB
5156          /                          & LDMDOR          ;
5157   003EF  /                          & DWRITE          ;
5158                     CONT             & AM2901 ,S0,SRAMQL,PASS,ZB
5159   003F0  /                          & SHIFT B#0110    ; RESTORE BYTE COUNT
5160                     CONT             & AM2901 ,S0,SRAMQR,SUBR,ZB
5161          /                          & CARRYH          ;
5162          /                          & LODMSR          ;
5163   003F1  /                          & SHIFT B#0110    ; BC=BC-1;S0=WC;MSR=BC
5164                     CONT             & AM2901 A,A,RAMF,OR,DA
5165   003F2  /                          & IMM H#8000      ; INC BA IN A
5166                     CONT             & AM2901 B,B,RAMF,ADD,DA
5167          /                          & CARRYL          ;
```

```
A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]     1:35 PM  TUE., 13  SEPT, 1983 PAGE 129
LINE    ADDR    STATEMENT

5168            /                           & IMM H#8001      ;
5169    003F3   /                           & LDMAR           ; INC BA IN B
5170                    CONT                & AM2901 S3,S3,RAMF,AND,DA
5171    003F4   /                           & IMM H#FF00      ;
5172                    CJP   SETMSR        & AM2901 ,S0,NOP,PASS,ZB
5173    003F5   /                           & CONDMSR NSGN    ; IF S0<0, END
5174                    CONT                & AM2901 ,A,RAMF,SUBR,ZB
5175    003F6   /                           & CARRYH          ; DEC A
5176                    JP    ENDEV         & AM2901 ,B,RAMF,SUBR,ZB
5177    003F7   /                           & CARRYH          ; DEC B
5178            SETMSR: JP    FILLCONT      & AM2901 ,S0,NOP,PASS,ZB
5179    003F8   /                           & LODMSR          ; SET MSR = COUNT STATUS

A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]     1:35 PM  TUE., 13  SEPT, 1983 PAGE 130
LINE    ADDR    STATEMENT

5181            ;
5182            ;           PATCH AREA
5183            ;
5184            INCS5:  RET             & AM2901 ,S5,RAMF,ADD,ZB
5185    003F9   /                       & CARRYH          ; INC S5
5186            ;
5187            VMAFPAT: JZ             & AM2901 ,A,RAMF,PASS,DZ
5188    003FA   /                       & IMM H#0050      ; A = ERROR 80 (DEC)
5189            ;
5190            .XFER:  CONT            & AM2901 BREL,A,RAMF,ADD,ZB
5191    003FB   /                       & CARRYL          ; BREL A
5192                    JP    XFCONT    & AM2901 BREL,B,RAMF,ADD,ZB
5193    003FC   /                       & CARRYL          ; BREL B

A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]     1:35 PM  TUE., 13  SEPT, 1983 PAGE 131
LINE    ADDR    STATEMENT

5195            ;*********************************************************************
5196            ;*                                                                   *
5197            ;*        SELF TEST JUMPS TO LAST WORD IN uSTORE                     *
5198            ;*                                                                   *
5199            ;*********************************************************************
5200            ;*****
5201    003FF           ORG H#3FF
5202    003FF   ;*****
5203            LASTWD: JP    INTPONOK  & AM2901 ,B,RAMF,ADD,ZB
5204    003FF   /                       & CARRYH          ; B := 1 FOR LED'S
```

```
LINE    ADDR    STATEMENT

5206            ;*********************************************************************
5207            ;*                                                                   *
5208            ;*        2K BASESET ROM AREA                                        *
5209            ;*                                                                   *
5210            ;*********************************************************************
5211            ;
5212            ;*********************************************************************
5213            ;*                                                                   *
5214            ;*        CDS Instructions                                           *
5215            ;*        ----------------                                           *
5216            ;*                                                                   *
5217            ;*        These instructions implement the stack                     *
5218            ;*        features of CDS.                                           *
5219            ;*                                                                   *
5220            ;*********************************************************************
5221            ;*****
5222            C.CQ:     CONT            & AM2901 CAB,Q,RAMF,EXOR,DA
5223            /                         & IMM H#8000      ; TEST CSMODE BIT
5224   00400    /                         & LUSRCOND        ; !LOADS WITH CARRY INVERT
5225                      CJP   SRG2NOP   & AM2901 CAB,Q,RAMF,AND,AB
5226            /                         & CONDUSR         ; !!SETCS == CONDUSR SGN
5227   00401    /                         & SPETC SETCS     ; LOAD Q; TURN ON CSMODE
5228            ;                                             IF CSMODE S.B. ON, JP
5229            C.CQ05:   JP    SRG2NOP   & AM2901 ,,NOP,ADD,AQ
5230   00402    /                         & SPETC CLRCS     ;  TURN OFF CSMODE; JP
5231            CCQ.:     CJP   FETCH     & AM2901 Q,CAB,RAMF,PASS,ZA
5232            /                         & CONDEXT CSON    ;
5233   00403    /                         & IFETCH          ; IF CSMODE ON, DONE
5234                      JZ              & AM2901 CAB,CAB,RAMF,OR,DA
5235   00404    /                         & IMM H#8000      ; OR IN CSMODE BIT
5236            C.Z:      JZ              & AM2901 CAB,ZREG,RAMF,PASS,ZA
5237   00405    /                         & IFETCH          ; PUT A/B IN Z ; FETCH
5238            CZ.:      JZ              & AM2901 ZREG,CAB,RAMF,PASS,ZA
5239   00406    /                         & IFETCH          ; PUT Z IN A/B; FETCH
5240            ADQ.:     JZ              & AM2901 Q,CAB,RAMF,ADD,AB
5241            /                         & CARRYL          ;
5242   00407    /                         & IFETCH          ;
5243            CIQ.:     CONT            & AM2901 ,,NOP,PASS,DZ
5244            /                         & IMM H#0020      ;
5245   00408    /                         & LDAER           ; POINT AER AT BOOT MEMORY
5246                      CONT            & AM2901 ,,NOP,PASS,DZ
5247            /                         & IMM IQLOC       ;
5248            /                         & LDMAR           ;
5249   00409    /                         & DREAD           ; READ INTERRUPTED_Q
5250                      CONT            & AM2901 MAPX,,NOP,PASS,DZ
5251   0040A    /                         & LDAER           ; PUT MAPX BACK IN AER
5252                      CONT            & AM2901 ,PC,NOP,SUBR,ZB
5253            /                         & CARRYH          ;
5254            /                         & LDMAR           ;
5255   0040B    /                         & IFETCH          ; FETCH NEXT INST
5256                      JZ              & AM2901 ,CAB,RAMF,PASS,DZ
5257   0040C    /                         & SPNOP           ; PUT IQ IN A/B
```

```
LINE    ADDR    STATEMENT

5259            ;*********************************************************************
5260            ;*                                                                   *
5261            ;*        .SDSP - Store display to memory                            *
5262            ;*                                                                   *
5263            ;*********************************************************************
5264            ;*****
5265            .SDSP:    CALL  INDRES     & AM2901 PC,S0,RAMA,PASS,DZ
5266            /                          & LODMSR          ; MSR = (DL=0)
5267            /                          & LDMAR           ;
5268   0040D    /                          & CREAD           ; S0 := DL; RESOLVE SECOND
5269                      CJP   TWIEXIT    & AM2901 Q,TAB,RAMF,PASS,ZA
5270            /                          & CONDMSR Z       ;
5271            /                          & LDMDOR          ;
5272   0040E    /                          & DWRITE          ; WRITE Q; IF DL=0, EXIT
5273                      CONT             & AM2901 ,S0,RAMF,SUBR,ZB
5274            /                          & CARRYH          ;
5275   0040F    /                          & LODMSR          ; DEC S0
5276                      CONT             & AM2901 Q,S1,RAMF,PASS,ZA
5277            /                          & LDMAR           ;
5278   00410    /                          & DREAD           ; READ STATIC_Q; S1 := Q
5279            SDSPL:    LDCT  $          & AM2901 ,,QREG,ADD,ZQ
5280            /                          & CARRYH          ;
5281   00411    /                          & LDMAR           ; INC DSP ADDRESS; LDMAR
5282                      CJP   TWIEXIT    & AM2901 TAB,S1,RAMF,PASS,DZ
5283            /                          & CONDMSR Z       ;
5284            /                          & LDMDOR          ;
5285   00412    /                          & DWRITE          ; WRITE STATIC_Q; IF R8=0,E
5286                      JRP   INTERRPT   & AM2901 S1,S0,RAMA,SUBR,ZB
5287            /                          & CARRYH          ;
5288            /                          & CONDEXT INTRPT  ;
5289            /                          & LODMSR          ;
5290            /                          & LDMAR           ;
5291   00413    /                          & DREAD           ; READ NEXT STATIC_Q
```

```
LINE    ADDR    STATEMENT

5293            ;*********************************************************************
5294            ;*                                                                   *
5295            ;*        PCAL - Procedure call                                      *
5296            ;*                                                                   *
5297            ;*        There are 5 versions of PCAL, all of which are             *
5298            ;*        very similar.                                              *
5299            ;*                                                                   *
5300            ;*        PCALI : Internal PCAL (to same segment)                    *
5301            ;*        PCALX : External PCAL (to possibly different segment)      *
5302            ;*        PCALV : Like PCALX, only the label to be called is in      *
5303            ;*                  data space instead of code space                 *
5304            ;*        PCALR : PCAL to old code, .ENTR compatible                 *
5305            ;*        PCALN : PCAL to old code, .ENTN compatible                 *
5306            ;*                                                                   *
5307            ;*********************************************************************
5308            ;*                                                                   *
5309            ;*        PCALI              +----> pe   DEC fs                      *
5310            ;*        DEF pe [,I] -------+                                       *
5311            ;*        DEC ac                                                     *
5312            ;*        DEF a_1                                                    *
5313            ;*        DEF a_2                                                    *
5314            ;*          :                                                        *
5315            ;*        DEF a_ac                                                   *
5316            ;*                                                                   *
5317            ;*        Builds a new frame on the stack of size FS                 *
5318            ;*        Copies AC parameter pointers to the new frame              *
5319            ;*          (AC may be 0)                                            *
5320            ;*        Executes next instruction at PE+1 in the current           *
5321            ;*          segment                                                  *
5322            ;*        This instruction is interruptible during parameter         *
5323            ;*          copying, and is restartable                              *
5324            ;*                                                                   *
5325            ;*********************************************************************
5326            .PCALI:   CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
5327            /                          & CARRYL          ;
5328            /                          & LDMAR           ;
5329   00414    /                          & CMSGN & DREAD   ;  RESOLVE PE; QREG := PE;
5330                      CONT             & AM2901 ,,NOP,ADD,AQ
5331   00415    /                          & CREAD           ; READ PE
5332                      CALL  PCASUB1    & AM2901 PC,S1,RAMA,SUBS,DZ
5333            /                          & CARRYL          ;
5334            /                          & LODMSR RESET    ;
5335            /                          & LDMAR           ;
5336   00416    /                          & CREAD           ; S1=FS-1; READ AC; MSR = 0
5337                      CALL  PCASUB2    & AM2901 ,S3,RAMF,ADD,ZB
5338            /                          & CARRYH          ;
5339   00417    /                          & LDMAR           ; MAR = NEXT Q ; S3 = NXT_Q
5340            PDONE:    CONT             & AM2901 S4,PC,RAMA,ADD,ZA
5341            /                          & CARRYH          ;
5342            /                          & LDMAR           ;
5343   00418    /                          & IFETCH          ; PC=PE+1 ; FETCH NEXT_INST
5344                      JZ               & AM2901 S2,Q,RAMF,SUBR,ZA
5345   00419    /                          & CARRYH          ; SET Q = NXT_Q ; JZ
```

```
LINE    ADDR    STATEMENT

5347            ;*********************************************************************
5348            ;*                                                                   *
5349            ;*        .PCALX                      External Segment               *
5350            ;*        LABEL PE  --------+                                        *
5351            ;*        DEC AC            +---->    PE   DEC FS                    *
5352            ;*        DEF A_0                                                    *
5353            ;*          :                                                        *
5354            ;*        DEF A_AC                                                   *
5355            ;*                                                                   *
5356            ;*********************************************************************
5357            ;*                                                                   *
5358            ;*        .PCALV            Code map            External segment   *
5359            ;*        DEF XL [,I] --+                                            *
5360            ;*        DEC AC        +-> XL  LABEL PE --+                         *
5361            ;*        DEF A_0                          +->  PE  DEC FS           *
5362            ;*          :                                                        *
5363            ;*        DEF A_AC                                                   *
5364            ;*                                                                   *
5365            ;*********************************************************************
5366            ;*                                                                   *
5367            ;*        LABEL PE is a label into the external segment              *
5368            ;*        being called.  The upper byte is a CST index               *
5369            ;*        and the lower byte is a STT index                          *
5370            ;*                                                                   *
5371            ;*        If the external segment is not mapped in, a fault to       *
5372            ;*        trap call SEGTRAP is executed.  The operating system       *
5373            ;*        is expected to place a JSB to the fault handling           *
5374            ;*        routine in this trap cell, and restart the PCAL            *
5375            ;*        when the new segment is in memory.                         *
5376            ;*                                                                   *
5377            ;*        PCALX(V) builds the stack frame in memory, copies          *
5378            ;*        the arguments, and then (after the instruction is          *
5379            ;*        no longer interruptible) maps in the new segment,          *
5380            ;*        determines PE and FS; writes NEXT_Q, sets PC and Q,        *
5381            ;*        and fetches the next instruction from PE+1                 *
5382            ;*                                                                   *
5383            ;*********************************************************************
5384            ;*****
5385            .PCALV:   CALL  READIND    & AM2901 BR,TABQ,QREG,ADD,DZ
5386            /                          & CARRYL         ;
5387            /                          & LODUSR         ;
5388            /                          & LDMAR          ;
5389   0041A    /                          & DREAD          ; RESOLVE, READ LABEL
5390            .PCALX:   CALL  RDCST      & AM2901 ,S1,RAMF,PASS,DZ
5391   0041B    /                          & SPNOP          ; S1 = LABEL ; READ CST ENT
5392                      CALL  PCAXSUB    & AM2901 ,S5,RAMF,PASS,DZ
5393            /                          & LODUSR         ;
5394   0041C    /                          & CREAD          ; S5=NEW CST ; READ CUR CS
5395            ;
5396            ;         IF SEGMENT NOT IN MEMORY, PCAXSUB RETURNS HERE
5397            ;
5398            EFAULT:   CONT             & AM2901 ,S7,RAMF,PASS,DZ
5399   0041D    /                          & IMM SEGTRAP    ; SEGMENT FAULT
```

```
LINE    ADDR    STATEMENT

5400            FAULT:    CJP   INTERRPT   & AM2901 ,,NOP,PASS,AQ
5401    0041E   /                          & CONDEXT INTRPT ; IF INT PEND, SERVICE IT
5402                      CALL  SETMAPS    & AM2901 ,,NOP,PASS,AQ
5403    0041F   /                          & SPNOP          ;
5404                      CONT             & AM2901 PC,PC,RAMF,SUBR,DA
5405            /                          & CARRYL         ;
5406    00420   /                          & IMM H#0002     ; RESTORE PC FOR FAULT
5407                      CONT             & AM2901 S7,CIR,NOP,PASS,ZA
5408            /                          & LDMAR          ;
5409    00421   /                          & IFETCH         ; FETCH FROM TRAP CELL
5410                      JZ               & AM2901 PC,PC,SRAMR,ADD,AB
5411            /                          & CARRYL         ;
5412    00422   /                          & SHIFT B#0000   ;
```

```
LINE    ADDR    STATEMENT

5414            ;*********************************************************************
5415            ;*                                                                   *
5416            ;*        PCALR, PCALN                                               *
5417            ;*                                                                   *
5418            ;*********************************************************************
5419            .PCALR:   CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
5420            /                          & CARRYL          ;
5421            /                          & CONDEXT MDIR15  ;
5422            /                          & LDMAR           ;
5423   00423    /                          & CMGO & DREAD    ;  RESOLVE PE
5424                      CALL  PCASUB1    & AM2901 PC,S4,RAMA,ADD,ZQ
5425            /                          & CARRYH          ;
5426            /                          & LDMAR           ;
5427   00424    /                          & CREAD           ; READ AC ; S4 = PE + 1
5428                      CONT             & AM2901 S2,S0,QREG,ADD,AB
5429   00425    /                          & CARRYL          ; QREG = NXT_Q+AC+1
5430                      CONT             & AM2901 ,S1,RAMF,ADD,DQ
5431            /                          & CARRYL          ;
5432   00426    /                          & IMM H#0008      ; S1 = NEXT_Q
5433                      CALL  CHECK      & AM2901 ZREG,S1,NOP,SUBR,AB
5434            /                          & CARRYL          ; TEST FOR STACK OVFL
5435   00427    /                          & LODUSR          ;
5436                      CONT             & AM2901 S2,S5,RAMF,ADD,DA
5437            /                          & CARRYL          ;
5438            /                          & IMM H#0004      ;
5439   00428    /                          & LDMAR           ; MAR = RESERVED WORD
5440                      CONT             & AM2901 ,S3,NOP,ADD,ZB
5441            /                          & CARRYH          ;
5442            /                          & LDMDOR          ;
5443   00429    /                          & DWRITE          ; WRITE TRUE RETURN ADDRESS
5444                      JP    WREXITS    & AM2901 ,S5,NOP,PASS,ZB
5445   0042A    /                          & LDMDOR          ;
5446            .PCALN:   CCALL INDRES     & AM2901 BR,TABQ,QREG,ADD,DZ
5447            /                          & CARRYL          ;
5448            /                          & CONDEXT MDIR15  ;
5449            /                          & LDMAR           ;
5450   0042B    /                          & CMGO & DREAD    ; RESOLVE PE
5451                      CALL  PCASUB1    & AM2901 PC,S4,RAMA,ADD,ZQ
5452            /                          & CARRYH          ;
5453            /                          & LDMAR           ;
5454   0042C    /                          & CREAD           ; READ AC; S4 = PE+1
5455                      CONT             & AM2901 S2,S0,QREG,ADD,AB
5456   0042D    /                          & CARRYL          ;
5457                      CONT             & AM2901 ,S1,RAMF,ADD,DQ
5458            /                          & CARRYL          ;
5459   0042E    /                          & IMM H#0008      ; S1 = NEW NEXT Q
5460                      CALL  CHECK      & AM2901 ZREG,S1,NOP,SUBR,AB
5461            /                          & CARRYL          ; TEST FOR STACK OVFL
5462   0042F    /                          & LODUSR          ;
5463                      CONT             & AM2901 S2,,NOP,ADD,DA
5464            /                          & CARRYL          ;
5465            /                          & IMM H#0005      ;
5466   00430    /                          & LDMDOR          ; MDOR = RETURN ADDRESS
```

```
LINE    ADDR    STATEMENT

5467            WREXITS:  CONT            & AM2901 ,S4,NOP,SUBR,ZB
5468            /                         & CARRYH          ;
5469            /                         & SPETC CLRCS     ;
5470            /                         & LDMAR           ;
5471    00431   /                         & DWRITE          ; WRITE RETURN ADDRESS
5472                      CONT            & AM2901 ,S3,RAMF,ADD,ZB
5473            /                         & CARRYH          ;
5474    00432   /                         & LDMAR           ;
5475                      CONT            & AM2901 ,,NOP,PASS,DZ
5476            /                         & IMM EXIT0       ;
5477            /                         & LDMDOR          ;
5478    00433   /                         & DWRITE          ; WRITE FIRST EXIT RETURN
5479                      CONT            & AM2901 ,S3,RAMF,ADD,ZB
5480            /                         & CARRYH          ;
5481    00434   /                         & LDMAR           ;
5482                      CONT            & AM2901 ,,NOP,PASS,DZ
5483            /                         & IMM EXIT1       ;
5484            /                         & LDMDOR          ;
5485    00435   /                         & DWRITE          ; WRITE SECOND EXIT RETURN
5486                      CONT            & AM2901 ,,NOP,PASS,DZ
5487            /                         & IMM EXIT2       ;
5488    00436   /                         & LDMDOR          ;
5489                      JP      PDONE   & AM2901 ,S3,RAMF,ADD,ZB
5490            /                         & CARRYH          ;
5491            /                         & LDMAR           ;
5492    00437   /                         & DWRITE          ; WRITE THIRD EXIT RETURN ;
5493            ;
5494            CHECK:    CJP     PCALMPR & AM2901 ,,NOP,ADD,AQ
5495    00438   /                         & CONDUSR NSGN    ; CHECK FOR MEMORY PROTECT
5496                      CJP     PCA00   & AM2901 PC,S7,RAMF,ADD,ZA
5497            /                         & CARRYH          ;
5498            /                         & CONDMSR NSGN    ;
5499            /                         & LDMAR           ;
5500    00439   /                         & CMGO & CREAD    ;  JP IF AC > 0
5501                      RET             & AM2901 S2,S3,RAMF,ADD,DA
5502            /                         & CARRYL          ;
5503    0043A   /                         & IMM H#0004      ; S3 = S2+4 = NXT_Q+5
```

```
LINE    ADDR    STATEMENT

5505            ;*********************************************************************
5506            ;*                                                                   *
5507            ;*        EXIT0 - RETURN FROM SUBROUTINE                             *
5508            ;*                                                                   *
5509            ;*        EXIT1 - RETURN WITH ONE SKIP                               *
5510            ;*        EXIT2 - RETURN WITH 2 SKIPS                                *
5511            ;*                                                                   *
5512            ;*********************************************************************
5513            .EXIT0:   CONT            & AM2901 Q,,NOP,ADD,DA
5514            /                         & CARRYL          ;
5515            /                         & IMM H#0002      ;
5516            /                         & LDMAR           ;
5517   0043B    /                         & DREAD           ; READ RETURN_P
5518                      CRET            & AM2901 Q,,NOP,ADD,ZA
5519            /                         & CARRYH          ;
5520            /                         & CONDMSR SGN     ;
5521   0043C    /                         & LDMAR           ; MAR = PREV_Q
5522                      CONT            & AM2901 ,S6,RAMF,PASS,DZ
5523   0043D    /                         & DREAD           ; S6 = RTN_P ; READ PREV_Q
5524            EXSKP:    CONT            & AM2901 ,S6,NOP,PASS,ZB
5525   0043E    /                         & LODUSR          ; USR = CROSS SEG STATUS
5526            ;                                           ;  WAIT FOR MEMORY
5527                      CCALL SGLD      & AM2901 ,S7,RAMF,PASS,DZ
5528   0043F    /                         & CONDUSR SGN     ; S7 = PREV_Q
5529                      CONT            & AM2901 S7,Q,RAMF,PASS,ZA
5530   00440    /                         & SPETC SETCS     ; SET Q; TURN CSMODE ON
5531                      CONT            & AM2901 S6,PC,RAMF,NOTRS,DA
5532            /                         & IMM H#8000      ;
5533            /                         & LDMAR           ;
5534   00441    /                         & IFETCH          ;
5535            INCPC:    JZ              & AM2901 ,PC,RAMF,ADD,ZB
5536   00442    /                         & CARRYH          ;
5537            .EXIT1:   CALL  .EXIT0    & AM2901 ,S6,RAMF,AND,ZQ
5538   00443    /                         & LODMSR SET      ; S6 = 0
5539            SKPEX:    JP    EXSKP     & AM2901 S6,S6,RAMF,ADD,DA
5540            /                         & CARRYH          ;
5541   00444    /                         & DREAD           ; READ PREV_Q ; S1 = RTN_P
5542            ;
5543            ;
5544            .EXIT2:   CALL  .EXIT0    & AM2901 ,S6,RAMF,AND,ZQ
5545   00445    /                         & LODMSR SET      ; S6 = 0 ; SET MSR FOR RETU
5546                      JP    SKPEX     & AM2901 ,S6,RAMF,ADD,ZB
5547   00446    /                         & CARRYH          ; S6 = 1
```

```
LINE    ADDR    STATEMENT

5549            ;*************************************************************************
5550            ;*                                                                       *
5551            ;*        PCAL SUBROUTINES                                               *
5552            ;*                                                                       *
5553            ;*************************************************************************
5554            NOARGS:   LDCT  PDONE      & AM2901 ZREG,S1,NOP,SUBR,AB
5555            /                          & CARRYL          ;
5556   00447    /                          & LODUSR          ; TEST FOR STACK OVERFLOW
5557                      JRP   PCALMPV    & AM2901 ,,NOP,ADD,AQ
5558   00448    /                          & CONDUSR NSGN    ; MP IF S1 >= Z
5559            ;
5560            ;         PCASUB - BUILDS STACK FRAME, MOVES PARAMETER ADDRESSES
5561            ;
5562            PCASUB1:  CONT             & AM2901 Q,,NOP,ADD,DA
5563            /                          & CARRYL          ;
5564            /                          & IMM H#0004      ;
5565   00449    /                          & LDMAR           ; MAR = NEXT_Q
5566                      CONT             & AM2901 TAB,S0,RAMF,PASS,DZ
5567   0044A    /                          & DREAD           ; S0 = AC ; READ NEXT_Q
5568                      CONT             & AM2901 Q,S4,RAMA,ADD,ZQ
5569            /                          & CARRYH          ;
5570   0044B    /                          & LDMDOR          ; MDOR = CURRENT Q ; S4 = P
5571                      CONT             & AM2901 ,S2,RAMF,ADD,DZ
5572            /                          & CARRYH          ;
5573            /                          & LDMAR           ;
5574   0044C    /                          & DWRITE          ; WRITE PREV_Q ; S2 = NXT_Q
5575                      CONT             & AM2901 S2,S3,RAMF,ADD,ZA
5576            /                          & CARRYH          ;
5577   0044D    /                          & LDMAR           ; MAR = RTN_P ; S3 = NXT_Q+
5578                      CONT             & AM2901 S0,PC,NOP,ADD,AB
5579            /                          & CARRYH          ;
5580            /                          & LDMDOR          ;
5581   0044E    /                          & DWRITE          ; WRITE RTN_P
5582                      CRET             & AM2901 S6,S3,RAMF,ADD,ZB
5583            /                          & CARRYH          ;
5584            /                          & CONDMSR SGN     ; RETURN IF FROM PCAXSUB
5585   0044F    /                          & LDMAR ;         ; MAR=AC ; S3=NXT_Q+3
5586                      RET              & AM2901 S0,S0,RAMA,SUBR,ZB
5587            /                          & CARRYH          ;
5588            /                          & LODMSR          ;
5589            /                          & LDMDOR          ;
5590   00450    /                          & DWRITE          ; WRITE AC ; S0 = MSR = AC-
```

```
LINE   ADDR   STATEMENT

5592           PCASUB2:  CJP   NOARGS     & AM2901 S2,S1,RAMF,ADD,AB
5593           /                          & CARRYL          ;
5594           /                          & CONDMSR SGN     ; IF AC=0, SKIP COPYARGS
5595           /                          & LDMDOR          ; S1 = MDOR = NEW NEXT Q
5596   00451   /                          & DWRITE          ; WRITE NNQ
5597                     CONT             & AM2901 ZREG,S1,NOP,SUBR,AB
5598           /                          & CARRYL          ;
5599   00452   /                          & LODUSR          ; TEST FOR STACK OVERFLOW
5600                     CJP   PCALMPV    & AM2901 PC,S7,RAMF,ADD,ZA
5601           /                          & CARRYH          ;
5602           /                          & CONDUSR NSGN    ;
5603           /                          & LDMAR           ;
5604   00453   /                          & CNMGO & CREAD   ; S1 = FROM ; READ FIRST DE
5605           PCA00:    CONT             & AM2901 S2,S3,RAMF,ADD,DA
5606           /                          & CARRYL          ;
5607           /                          & IMM H#0005      ;
5608   00454   /                          & LDMAR           ; S3 = TO = NXT_Q+6
5609           COPYARGS: CJP   PINDR      & AM2901 BR,TABQ,NOP,ADD,DZ
5610           /                          & CARRYL          ;
5611           /                          & CONDEXT MDIR15  ; IF MDIR15: JP, LDMAR, DRE
5612   00455   /                          & SPIND           ;  ELSE: CONT, LDMDOR, DWRI
5613                     CRET             & AM2901 S7,S7,RAMF,ADD,ZB
5614           /                          & CARRYH          ; INCREMENT FROM ADDRESS
5615   00456   /                          & CONDMSR Z       ;  DONE IF MSR Z
5616                     CJP   INTERRPT   & AM2901 S7,S0,RAMA,SUBR,ZB
5617           /                          & CARRYH          ;
5618           /                          & CONDEXT INTRPT  ;
5619           /                          & LODMSR          ;
5620           /                          & LDMAR           ;
5621   00457   /                          & CNMGO & CREAD   ;  DECREMENT S0 ; READ NEXT
5622           PCA05:    JP    COPYARGS   & AM2901 ,S3,RAMF,ADD,ZB
5623           /                          & CARRYH          ;
5624   00458   /                          & LDMAR           ; MAR = S3 = NEXT TO
5625           ;
5626           PINDR:    CJP   COPYARGS   & AM2901 ,S3,NOP,PASS,ZB
5627           /                          & CONDEXT NOABIN  ;
5628   00459   /                          & LDMAR           ; CHECK FOR INTS OR A/B
5629                     CJP   INTERRPT   & AM2901 ,,NOP,ADD,AQ
5630   0045A   /                          & CONDEXT INTRPT  ; IF INT PEND
5631                     CONT             & AM2901 TAB,,NOP,PASS,DZ
5632           /                          & LDMDOR          ;
5633   0045B   /                          & DWRITE          ; WRITE A/B TO MEMORY
5634                     JP    PCA05      & AM2901 ,S3,RAMF,SUBR,ZB
5635           /                          & CARRYH          ;
5636   0045C   /                          & DREAD           ; GET IT IN THE MDIR
```

```
LINE    ADDR    STATEMENT

5638            ;
5639            ;       RDCST - S5 = CST ENTRY FOR NEW SEGMENT
5640            ;               S6 = CST ENTRY FOR THIS SEGMENT
5641            ;               IF NEW SEGMENT NOT IN MEMORY, RETURN TO PFAULT
5642            ;                 ELSE CALL PCASUB
5643            ;
5644            RDCST:  CONT            & AM2901 S1,S2,RAMA,PASS,DZ
5645    0045D   /                       & SPRD RL4      ; EXTRACT NEW CST FROM LABE
5646                    PUSH  H#001     & AM2901 S2,S2,RAMA,PASS,DZ
5647    0045E   /                       & SPRD RL4      ;
5648                    RFCT            & AM2901 ,S2,SRAML,PASS,ZB
5649    0045F   /                       & SHIFT B#0010  ; S2=ADDRESS OF NEW CST ENT
5650                    CONT            & AM2901 S2,S2,RAMF,AND,DA
5651            /                       & IMM H#01FC    ;
5652            /                       & LDMAR         ;
5653    00460   /                       & CREAD         ; READ NEW CST ENTRY
5654                    RET             & AM2901 ,,NOP,PASS,DZ
5655            /                       & IMM H#0400    ;
5656    00461   /                       & LDMAR         ; MAR = CURRENT CST
5657            ;
5658            ;
5659            PCAXSUB:CRET            & AM2901 PC,PC,SRAMR,ADD,AB
5660            /                       & CARRYL        ; SET SIGN BIT OF PC FOR
5661            /                       & CONDUSR SGN   ; PCASUB.  IF SIGN BIT OF N
5662    00462   /                       & SHIFT B#0001  ; CST SET, RTN (FAULT)
5663                    CALL  PCASUB1   & AM2901 PC,S6,RAMA,PASS,DZ
5664            /                       & LODMSR SET    ;
5665            /                       & LDMAR         ;
5666    00463   /                       & CREAD         ; S6=CUR CST ; MSR=1 ; READ
5667            ;
5668            ;       WRITE RETURN CST AND ARG_COUNT
5669            ;
5670                    CONT            & AM2901 S6,S6,RAMF,AND,DA
5671    00464   /                       & IMM H#FF00    ; MASK CUR CST TO 8 BITS
5672                    CONT            & AM2901 S6,S0,NOP,OR,AB
5673            /                       & LDMDOR        ;
5674    00465   /                       & DWRITE        ; WRITE RTN CST AND AC
5675                    CONT            & AM2901 ,S0,RAMF,SUBR,ZB
5676            /                       & CARRYH        ;
5677    00466   /                       & LODMSR        ; DEC COUNT ; MSR = COUNT
5678                    CCALL PCA00     & AM2901 PC,S7,RAMF,ADD,ZA
5679            /                       & CARRYH        ;
5680            /                       & CONDMSR NSGN  ;
5681            /                       & LDMAR         ;
5682    00467   /                       & CMGO & CREAD  ;  IF AC>0, COPYARGS
5683                    CONT            & AM2901 ,S4,RAMF,AND,ZQ
5684    00468   /                       & LODMSR RESET  ; S4 = 0
```

```
LINE    ADDR    STATEMENT

5686            MAPIT:    CONT            & AM2901 S5,S5,RAMF,OR,DA
5687    00469   /                         & IMM H#4000      ; S5=INIT PMR VALUE
5688            ;                                             WITH WRITE PROTECT BIT SE
5689                      PUSH H#01E        & AM2901 MAPX,,NOP,ADD,DZ
5690            /                         & CARRYH          ;
5691    0046A   /                         & LDAER           ; AER POINTS TO CODE MAP
5692                      CONT              & AM2901 S4,S4,RAMF,ADD,DA
5693            /                         & CARRYL          ;
5694            /                         & IMM H#0400      ;
5695    0046B   /                         & LDMAR           ; MAR -> MAP NUMBER
5696                      RFCT              & AM2901 S5,S5,RAMA,ADD,ZB
5697            /                         & CARRYH          ;
5698    0046C   /                         & SPWR MAPWR      ; WRITE A MAP, LOOP
5699                      CRET              & AM2901 MAPX,,NOP,PASS,DZ
5700            /                         & CONDMSR SGN     ; SGLD EXITS HERE
5701    0046D   /                         & LDAER           ; AER -> EXECUTE MAP
5702                      CONT              & AM2901 S1,S1,RAMF,NOTRS,DA
5703    0046E   /                         & IMM H#FF00      ;
5704                      CONT              & AM2901 S1,,NOP,ADD,DA
5705            /                         & CARRYL          ;
5706            /                         & IMM H#0401      ;
5707            /                         & LDMAR           ;
5708    0046F   /                         & CREAD           ; READ STT ENTRY
5709                      CONT              & AM2901 S2,S3,RAMF,ADD,DA
5710            /                         & CARRYL          ;
5711    00470   /                         & IMM H#0003      ; S3 = NEXT_Q
5712                      CONT              & AM2901 ,S4,RAMF,PASS,DZ
5713            /                         & LDMAR           ;
5714    00471   /                         & CREAD           ; S4=STT ENTRY ; READ FS
5715                      CONT              & AM2901 ,S4,RAMF,ADD,ZB
5716    00472   /                         & CARRYH          ; S4 = INIT PC
5717                      CONT              & AM2901 S3,S1,RAMA,SUBS,DZ
5718            /                         & CARRYL          ;
5719    00473   /                         & LDMAR           ; S1=FS-1 ; MAR = NEXT_Q AD
5720                      JP     NOARGS     & AM2901 S2,S1,RAMF,ADD,AB
5721            /                         & CARRYL          ;
5722            /                         & LDMDOR          ;
5723    00474   /                         & DWRITE          ; WRITE NEXT_Q
```

```
LINE    ADDR    STATEMENT

5725            ;
5726            PCALMPR:  CONT            & AM2901 ,,NOP,ADD,AQ
5727    00475   /                         & SPETC CLRCS     ; TURN CSMODE OFF FOR PCALR
5728            PCALMPV:  CJP   GENMPV    & AM2901 S2,Q,RAMF,SUBR,ZA
5729            /                         & CARRYH          ;
5730    00476   /                         & CONDEXT MPEN    ;  IF MP SYS ON, GENMPV
5731                      CONT            & AM2901 PC,S7,RAMF,SUBR,ZA
5732            /                         & CARRYH          ;
5733    00477   /                         & ENCN SETMPEN    ;  TURN MP SYS ON
5734                      CONT            & AM2901 ,S7,RAMF,SUBR,ZB
5735            /                         & CARRYH          ;
5736            /                         & LDMAR           ;
5737    00478   /                         & IFETCH          ;  REFETCH INST WITH MP ON
5738                      JP    GENMPV    & AM2901 ,,NOP,ADD,AQ
5739    00479   /                         & ENCN CLRMPEN    ; TURN MP BACK OFF, GENMPV
5740            ;*****************************************************************
5741            ;*                                                               *
5742            ;*        SGLD - MAP IN SEGMENT TO RETURN TO FOR EXIT            *
5743            ;*                                                               *
5744            ;*****************************************************************
5745            ;
5746            SGLD:     CONT            & AM2901 Q,,NOP,ADD,DA
5747            /                         & CARRYL          ;
5748            /                         & IMM H#0003      ;
5749            /                         & LDMAR           ;
5750    0047A   /                         & DREAD           ; READ RTN_CST
5751                      CONT            & AM2901 ,,NOP,ADD,AQ
5752    0047B   /                         & SPNOP           ; WAIT FOR DATA
5753                      CALL  RDCST     & AM2901 ,S1,RAMF,PASS,DZ
5754    0047C   /                         & LODMSR SET      ; S1 = RTN_CST | AC; MSR=1
5755                      LDCT  EFAULT    & AM2901 ,S5,RAMF,PASS,DZ
5756    0047D   /                         & LODUSR          ; S5 = CST_ENTRY ; MSR = 1
5757                      JRP   MAPIT     & AM2901 ,S4,RAMF,AND,ZQ
5758    0047E   /                         & CONDUSR NSGN    ;  S4=0;SEG IN MEM IF SIGN
```

```
LINE    ADDR    STATEMENT

5760            ;*****************************************************************
5761            ;       Thunder Single Precision Floating Point Microcode       *
5762            ;                                                               *
5763            ;  The operands are manipulated in unpacked format:  two words  *
5764            ;  of mantissa and one word of (two's complement) exponent.     *
5765            ;  The names are:                                               *
5766            ;                                                               *
5767            ;     1st argument:  (XU,XL,XEXP) => result                     *
5768            ;     2nd argument:  (YU,YL,YEXP)         scratch: ZU,ZL        *
5769            ;*****************************************************************
5770            ;
5771            XU:       EQU A   (A)         ; REGISTER ASSIGNMENTS.
5772            XL:       EQU B   (B)
5773            XEXP:     EQU S1
5774            ;
5775            YU:       EQU DBLIMSW
5776            YL:       EQU S2
5777            YEXP:     EQU DBLILSW
5778            ;*****
5779            ZU:       EQU S5
5780            ZL:       EQU S6
```

```
LINE    ADDR    STATEMENT

5782            ;*****************************************************************
5783            ;                                                               *
5784            ;       FIX:  Floating to single integer conversion.            *
5785            ;                                                               *
5786            ;       convert (A,B) from floating to integer; A := result.    *
5787            ;                                                               *
5788            ;       If the exponent exceeds +15, the result is 77777B       *
5789            ;       and overflow is set.  If the exponent is less than 0,   *
5790            ;       the result is zero.                                     *
5791            ;                                                               *
5792            ;*****************************************************************
5793            ;
5794            ;       UNPACK.  (SEE CALLING SEQ. FOR UNPACK)
5795            ;       IF XEXP < 0, RESULT = 0.
5796            ;       XEXP := 15-XEXP
5797            ;       Q := 0
5798            ;       IF XEXP < 0, OVERFLOW.
5799            ;
5800            FIX:    CALL  UNPACK1   & AM2901 B,XEXP,RAMF,PASS,ZA
5801            /                       & LODMSR SWAPEO  ; UNPACK <A,B>
5802   0047F    /                       & ENBLC & ENBLO  ;
5803                    CJP   FIXZERO   & AM2901 ,,QREG,AND,ZQ
5804            /                       & CONDMSR SGN    ;
5805   00480    /                       & CMGO & IFETCH  ; Q = 0; IF XEXP<0, ZERO
5806                    CONT            & AM2901 XEXP,XEXP,RAMF,SUBS,DA
5807            /                       & CARRYH         ;
5808            /                       & LUSRCOND       ;
5809   00481    /                       & IMM H#000F     ; XEXP=15-XEXP
5810                    CJP   FIXOVR    & AM2901 ,XEXP,RAMF,SUBR,ZB
5811            /                       & CARRYH         ;
5812            /                       & CONDUSR SGN    ;
5813   00482    /                       & CMGO & IFETCH  ; IF XEXP > 15, OVERFLOW
5814                    CJP   FIXDONE   & AM2901 ,XU,NOP,PASS,ZB
5815            /                       & CONDUSR SGN    ;
5816   00483    /                       & CMGO & IFETCH  ; IF XEXP WAS 15, NO SHIFT
5817                    PUSH            & AM2901 ,XEXP,NOP,PASS,ZB
5818            /                       & SPETC LDCTY    ;
5819   00484    /                       & IFETCH         ; 2910 = SHIFT COUNT
5820                    RFCT            & AM2901 ,XU,SRAMQR,ADD,ZB
5821            /                       & CARRYL         ;
5822            /                       & LODUSR         ;
5823   00485    /                       & SHIFT B#1110   ; SIGN EXTEND
5824            ;
5825            ;       DONE SHIFTING.  IF VALUE IS NEGATIVE, AND XL<>0 OR ANY BITS
5826            ;       WERE SHIFTED INTO Q, ADD ONE TO THE FIXED VALUE.
5827            ;
5828            FIXDONE: CJP  FETCH     & AM2901 XL,,NOP,OR,AQ
5829            /                       & CONDUSR NSGN   ;
5830   00486    /                       & IFETCH         ; IF NOT NEGATIVE
5831                    JZ              & AM2901 ,XU,RAMF,ADD,ZB
5832            /                       & CARRYEXT       ;
5833   00487    /                       & CONDUSR NZ     ; ADD ONE IF B<>0 OR Q<>0
5834            ;
```

```
LINE    ADDR    STATEMENT

5835            ;         EXPONENT NEGATIVE, RESULT = 0
5836            ;
5837            FIXZERO:  JZ              & AM2901 ,XU,RAMF,AND,ZB
5838   00488    /                         & SPNOP          ; XU (A) = 0
5839            ;
5840            ;         EXPONENT > 15, OVERFLOW (A=77777,O=1)
5841            ;
5842            FIXOVR:   JZ              & AM2901 A,A,SRAMR,EXNOR,AB
5843            /                         & LODMSR SET     ;
5844            /                         & ENBLO          ;
5845   00489    /                         & SHIFT B#0000   ;
```



```
LINE    ADDR    STATEMENT

5847            ;*****************************************************************
5848            ;                                                                *
5849            ;         'FLT' SINGLE INTEGER TO FLOATING-POINT CONVERSION.     *
5850            ;                                                                *
5851            ;         CONVERT (A) FROM INTEGER TO FLOATING; (A,B) = RESULT.  *
5852            ;                                                                *
5853            ;         NO UNDERFLOW, OVERFLOW OR ERROR CONDITIONS CAN OCCUR.  *
5854            ;                                                                *
5855            ;*****************************************************************
5856            ;
5857            ;         XU := A  (BY DEFINITION)
5858            ;         XEXP := 15
5859            ;         XL := 0
5860            ;         GO NORMALIZE & PACK.
5861            ;
5862            FLT:      CONT            & AM2901 ,XEXP,RAMF,PASS,DZ
5863   0048A    /                         & IMM H#000F     ;     XEXP = 15
5864                      JP    NORM      & AM2901 ,XL,RAMF,AND,ZB
5865   0048B    /                         & SPNOP          ;     XL = 0
```

```
LINE    ADDR    STATEMENT

5867            ;*********************************************************************
5868            ;*                                                                   *
5869            ;*        Fix single precision to double integer.                    *
5870            ;*                                                                   *
5871            ;*        JSB .FIXD                                                  *
5872            ;*                                                                   *
5873            ;*********************************************************************
5874            ;
5875            ;         If exponent < 0, result = 0; else use .TFXD
5876            ;
5877            .FIXD:    LDCT  TFXD1      & AM2901 A,S4,RAMF,PASS,ZA
5878            /                          & LODMSR SWAPEO  ;
5879   0048C    /                          & ENBLC & ENBLO  ;
5880                      CONT             & AM2901 B,S5,RAMF,AND,DA
5881   0048D    /                          & IMM H#FF00     ;
5882                      CONT             & AM2901 S5,B,SRAMR,EXOR,AB
5883   0048E    /                          & SHIFT B#0010   ;
5884                      CJP   FIXD1      & AM2901 ,S6,RAMF,AND,ZQ
5885   0048F    /                          & CONDMSR NC     ;
5886                      JP    NGL02      & AM2901 ,A,RAMF,AND,ZQ
5887            /                          & LODMSR SWAPEO  ;
5888            /                          & ENBLC & ENBLO  ;
5889   00490    /                          & IFETCH         ;
5890            FIXD1:    CONT             & AM2901 B,MP,RAMF,SUBS,DA
5891            /                          & CARRYH         ;
5892            /                          & IMM H#000F     ;
5893   00491    /                          & LUSRCOND       ;
5894                      JRP   TFXD4      & AM2901 ,PC,RAMF,SUBR,ZB
5895            /                          & CARRYH         ;
5896   00492    /                          & CONDUSR SLE    ;
```

```
LINE    ADDR    STATEMENT

5898            ;*****************************************************************
5899            ;*                                                               *
5900            ;*       Float double integer to single precision.               *
5901            ;*                                                               *
5902            ;*****************************************************************
5903            ;
5904            .FLTD:    LDCT  DNCOUNT1   & AM2901 ,A,NOP,OR,ZB
5905            /                          & CARRYH         ;
5906            /                          & LODMSR SWAPEO  ;
5907    00493   /                          & ENBLC & ENBLO  ;
5908                      CJP   FLTD1      & AM2901 A,B,NOP,EXOR,AB
5909    00494   /                          & CONDUSR UGT    ; IF NOT ZERO OR ONES
5910                      CJP   FLTD2      & AM2901 ,B,NOP,PASS,ZB
5911    00495   /                          & CONDUSR NSGN   ;
5912            ;
5913            ;         Outside [-32768,+32767]; float (A,B) * 2**16
5914            ;
5915            FLTD1:    CONT             & AM2901 ,MP,RAMF,PASS,DZ
5916    00496   /                          & IMM H#0020     ;
5917                      JP    FLTD3      & AM2901 ,B,QREG,PASS,ZB
5918    00497   /                          & LODMSR RESET   ;
5919            ;
5920            ;         Inside [-32768,+32767]; float (B,0) * 2**16
5921            ;         If zero, just exit.
5922            ;
5923            FLTD2:    CJP   CLRO       & AM2901 B,A,RAMF,PASS,ZA
5924            /                          & CONDUSR Z      ;
5925    00498   /                          & CMGO & IFETCH  ;
5926                      CONT             & AM2901 ,,QREG,AND,ZQ
5927    00499   /                          & LODMSR RESET   ; QREG = 0
5928                      CONT             & AM2901 ,MP,RAMF,PASS,DZ
5929    0049A   /                          & IMM H#0010     ;
```

```
LINE    ADDR    STATEMENT

5931            ;       Normalize and pack (A,Q) * 2**R12
5932            ;
5933            FLTD3:  CPUSH           & AM2901 ,A,SRAMQL,PASS,ZB
5934            /                       & CONDMSR SGN     ; NEVER LOAD COUNTER
5935    0049B   /                       & SHIFT B#0100    ;
5936                    TWB             & AM2901 ,A,SRAMQL,PASS,ZB
5937            /                       & CONDEXT IRSKBF  ;
5938    0049C   /                       & SHIFT B#0100    ;
5939                    CONT            & AM2901 ,A,SRAMQR,PASS,ZB
5940            /                       & LODMSR INVERT   ;
5941            /                       & ENBLC           ;
5942    0049D   /                       & SHIFT B#0100    ;
5943                    JSRP            & AM2901 ,A,SRAMQR,PASS,ZB
5944            /                       & CONDMSR NSGN    ;
5945    0049E   /                       & SHIFT B#0100    ;
5946                    CONT            & AM2901 MP,S7,SRAML,SUBS,AB
5947            /                       & CARRYH          ;
5948            /                       & LODMSR RESET    ;
5949            /                       & ENBLC           ; RESET MC (=MO AFTER SWAP)
5950    0049F   /                       & SHIFT B#0000    ;
5951                    CONT            & AM2901 S7,B,RAMF,AND,DQ
5952            /                       & IMM H#FF00      ;
5953    004A0   /                       & IFETCH          ;
5954                    JZ              & AM2901 S7,B,RAMF,OR,AB
5955            /                       & LODMSR SWAPEO   ;
5956    004A1   /                       & ENBLC & ENBLO   ;
```

```
LINE    ADDR    STATEMENT

5958            ;*****************************************************************
5959            ;                                                                *
5960            ;         '.FSB' FLOATING-POINT SUBTRACT.                        *
5961            ;                                                                *
5962            ;         SUBTRACT THE UNPACKED NUMBERS                          *
5963            ;                                                                *
5964            ;             (XU,XL,XEXP) - (YU,YL,YEXP) --> (XU,XL,XEXP)       *
5965            ;                                                                *
5966            ;         OVERFLOW AND UNDERFLOW ARE POSSIBLE.  THE RESULT       *
5967            ;         MAY BE UNNORMALIZED.                                   *
5968            ;                                                                *
5969            ;*****************************************************************
5970            ;
5971            ;         READ SECOND OPERAND FROM MEMORY; UNPACK BOTH OPERANDS.
5972            ;         YL := -YL
5973            ;         YU := -YU - BORROW
5974            ;         IF OVERFLOW,
5975            ;           YU := YU RS 1
5976            ;           YEXP := YEXP + 1
5977            ;         MERGE INTO .FAD CODE.
5978            ;
5979            .FSB:     CALL  DIARG      & AM2901 BR,TABQ,QREG,ADD,DZ
5980            /                          & CARRYL          ;
5981            /                          & LODUSR          ;
5982            /                          & LDMAR           ;
5983    004A2   /                          & DREAD           ;
5984                      CALL  UNPACK     & AM2901 XL,XEXP,RAMF,PASS,ZA
5985            /                          & LODMSR SWAPEO   ;
5986    004A3   /                          & ENBLC & ENBLO   ;
5987                      CONT             & AM2901 ,YL,RAMF,SUBS,ZB
5988            /                          & CARRYH          ;
5989    004A4   /                          & LODUSR          ;
5990                      CONT             & AM2901 ,YU,RAMF,SUBS,ZB
5991    004A5   /                          & CARRYUC         ;
5992                      CJP   FADD1      & AM2901 ,,NOP,ADD,AQ
5993    004A6   /                          & CONDUSR NOVR    ;
5994                      CONT             & AM2901 ,YU,SRAMR,PASS,ZB
5995    004A7   /                          & SHIFT B#0000    ;
5996                      JP    FADD1      & AM2901 ,YEXP,RAMF,ADD,ZB
5997    004A8   /                          & CARRYH          ;
```

```
LINE    ADDR    STATEMENT

5999            ;*****************************************************************
6000            ;                                                               *
6001            ;       '.FAD' FLOATING-POINT ADD.                              *
6002            ;                                                               *
6003            ;       ADD THE UNPACKED NUMBERS                                *
6004            ;                                                               *
6005            ;           (XU,XL,XEXP) + (YU,YL,YEXP) --> (XU,XL,XEXP)        *
6006            ;                                                               *
6007            ;       OVERFLOW AND UNDERFLOW ARE POSSIBLE.  THE RESULT        *
6008            ;       MAY BE UNNORMALIZED.                                    *
6009            ;                                                               *
6010            ;*****************************************************************
6011            ;
6012            ;       READ SECOND OPERAND FROM MEMORY; UNPACK BOTH OPERANDS.
6013            ;
6014            .FAD:   CALL  DIARG     & AM2901 BR,TABQ,QREG,ADD,DZ
6015            /                       & CARRYL        ;
6016            /                       & LODUSR        ;
6017            /                       & LDMAR         ;
6018    004A9   /                       & DREAD         ; RESOLVE, READ Y
6019                    CALL  UNPACK    & AM2901 XL,XEXP,RAMF,PASS,ZA
6020            /                       & LODMSR SWAPEO ;
6021    004AA   /                       & ENBLC & ENBLO ;
6022            ;
6023            ;       ZU := XU
6024            ;       IF XU=0 THEN SWAP (XU,XL)<=>(YU,YL) & SET XEXP := YEXP
6025            ;       IF YU=0 THEN DONE.
6026            ;       Q := (XEXP-YEXP); TEST IT.
6027            ;
6028            FADD1:  CONT            & AM2901 XU,ZU,RAMF,PASS,ZA
6029    004AB   /                       & LODUSR        ;
6030                    CCALL SWAPFARG  & AM2901 YEXP,YU,NOP,PASS,ZB
6031    004AC   /                       & CONDUSR Z     ; IF XU = 0, SWAP
6032                    CJP   NORM      & AM2901 YEXP,XEXP,QREG,SUBR,AB
6033            /                       & CARRYL        ;
6034    004AD   /                       & CONDUSR Z     ;
6035            ;
6036            ;       ZL := XL
6037            ;       IF Q < 0 THEN
6038            ;         SWAP (XU,XL)<=>(YU,YL)   (USING ZU,ZL)
6039            ;         XEXP := YEXP
6040            ;         Q := -Q
6041            ;
6042                    CCALL SWAPFARG  & AM2901 XL,ZL,RAMF,PASS,ZA
6043    004AE   /                       & CONDUSR SGN   ;
```

```
LINE    ADDR    STATEMENT

6045            ;       YEXP := Q   (SHIFT COUNT); TEST IT.
6046            ;       TEST (YEXP-25)
6047            ;       Q := YL
6048            ;       IF (YEXP-25) >= 0, DONE (SWAMP).
6049            ;         ELSE GOTO FADD3 (IN COUNTER)
6050            ;
6051                    CONT            & AM2901 ,YEXP,RAMF,PASS,ZQ
6052    004AF   /                       & LODUSR        ; YEXP = SHIFT COUNT
6053                    CJP   FADD2     & AM2901 YEXP,YL,QREG,PASS,ZB
6054    004B0   /                       & CONDUSR Z     ; IF SHIFT COUNT = 0, JP
6055                    CONT            & AM2901 YEXP,,NOP,SUBR,DA
6056            /                       & CARRYL        ;
6057            /                       & LUSRCOND      ;
6058    004B1   /                       & IMM H#0019    ; SUB 25
6059                    CJP   NORM      & AM2901 YEXP,ZL,NOP,ADD,AQ
6060    004B2   /                       & CONDUSR NSGN  ; Q=YL; IF SWAMP, DONE
6061                    CONT            & AM2901 YEXP,ZL,RAMF,SUBR,DA
6062            /                       & CARRYL        ;
6063            /                       & LUSRCOND      ;
6064    004B3   /                       & IMM H#0010    ; TEST FOR SHCT>15
6065                    CJP   ALIGN     & AM2901 ,YU,NOP,PASS,ZB
6066    004B4   /                       & CONDUSR SGN   ; JP IF SHCT<=15
6067                    CONT            & AM2901 YU,YU,QREG,PASS,ZB
6068    004B5   /                       & LODMSR SWAPUM ; QREG=YU
6069                    CONT            & AM2901 YU,YU,RAMF,SUBR,AB
6070            /                       & CARRYEXT      ;
6071    004B6   /                       & CONDMSR SGN   ; SIGN EXTEND IN YU
6072                    CJP   FADD2     & AM2901 ,ZL,NOP,PASS,ZB
6073            /                       & CONDUSR Z     ;
6074    004B7   /                       & SEQFRZ        ; IF SHCT = 16
6075            ;
6076            ;       SHIFT (YU,Q) RIGHT BY (YEXP) BITS:
6077            ;
6078            ;       LOAD 2910 COUNTER WITH SHIFT COUNT AND
6079            ;       ARITHMETICALLY SHIFT RIGHT (CNTR) TIMES
6080            ;
6081            ALIGN:  PUSH            & AM2901 ,YEXP,NOP,SUBR,ZB
6082            /                       & CARRYH        ;
6083    004B8   /                       & SPETC LDCTY   ; LOAD 2910 COUNTER
6084                    RFCT            & AM2901 ,YU,SRAMQR,ADD,ZB
6085            /                       & CARRYL        ;
6086    004B9   /                       & SHIFT B#1110  ; ARITH SHIFT
```

```
LINE    ADDR    STATEMENT

6088            ;       (XU,Q) := (XU,XL) + (YU,Q)
6089            ;       IF OVERFLOW,
6090            ;         (XU,Q) := (XU,Q) RS 1
6091            ;         XEXP := XEXP + 1
6092            ;       XL := Q
6093            ;       GO NORMALIZE & PACK.
6094            ;
6095            FADD2:  CONT            & AM2901 XL,,QREG,ADD,AQ
6096            /                       & CARRYL        ;
6097    004BA   /                       & LODUSR        ;
6098                    CONT            & AM2901 YU,XU,RAMF,ADD,AB
6099    004BB   /                       & CARRYUC       ;
6100                    CJP   NORM      & AM2901 ,XL,RAMF,PASS,ZQ
6101    004BC   /                       & CONDUSR NOVR  ;
6102                    CONT            & AM2901 ,XU,SRAMQR,ADD,ZB
6103            /                       & CARRYL        ;
6104    004BD   /                       & SHIFT B#1110  ;
6105                    CONT            & AM2901 XU,XU,RAMF,EXOR,DA
6106    004BE   /                       & IMM H#8000    ;
6107                    CONT            & AM2901 ,XEXP,RAMF,ADD,ZB
6108    004BF   /                       & CARRYH        ;
6109                    JP    NORM      & AM2901 ,XL,RAMF,PASS,ZQ
6110    004C0   /                       & SPNOP         ;
6111            ;       ROUTINE TO SWAP (XU,XL) AND (YU,YL)  (GIVEN ZU,ZL SET UP)
6112            ;       ALSO:   XEXP := YEXP     Q := -Q
6113            ;       uZ = ZERO CONDITION FOR NEW YU
6114            ;
6115            SWAPFARG: CONT          & AM2901 YU,XU,RAMF,PASS,ZA
6116    004C1   /                       & SPNOP         ;
6117                    CONT            & AM2901 YL,XL,RAMF,PASS,ZA
6118    004C2   /                       & SPNOP         ;
6119                    CONT            & AM2901 YEXP,XEXP,RAMF,PASS,ZA
6120    004C3   /                       & SPNOP         ;
6121                    CONT            & AM2901 ZL,YL,RAMF,PASS,ZA
6122    004C4   /                       & SPNOP         ;
6123                    CONT            & AM2901 ,,QREG,SUBS,ZQ
6124    004C5   /                       & CARRYH        ;
6125                    RET             & AM2901 ZU,YU,RAMF,PASS,ZA
6126    004C6   /                       & LODUSR        ;
```

```
LINE    ADDR    STATEMENT

6128            ;*****************************************************************
6129            ;                                                               *
6130            ;       '.FMP' FLOATING-POINT MULTIPLY.                         *
6131            ;                                                               *
6132            ;       MULTIPLY THE UNPACKED NUMBERS                           *
6133            ;                                                               *
6134            ;           (XU,XL,XEXP) * (YU,YL,YEXP) --> (XU,XL,XEXP)        *
6135            ;                                                               *
6136            ;       OVERFLOW AND UNDERFLOW ARE POSSIBLE.  THE RESULT        *
6137            ;       MAY BE UNNORMALIZED BY ONE BIT.                         *
6138            ;                                                               *
6139            ;*****************************************************************
6140            ;
6141            ;       READ SECOND OPERAND FROM MEMORY; UNPACK BOTH OPERANDS.
6142            ;       XEXP := XEXP + YEXP + 1
6143            ;
6144            .FMP:   CALL  DIARG     & AM2901 BR,TABQ,QREG,ADD,DZ
6145            /                       & CARRYL        ;
6146            /                       & LODUSR        ;
6147            /                       & LDMAR         ;
6148    004C7   /                       & DREAD         ; GET Y ARG
6149                    CALL  UNPACK    & AM2901 XL,XEXP,RAMF,PASS,ZA
6150            /                       & LODMSR SWAPEO ;
6151    004C8   /                       & ENBLC & ENBLO ;
6152            ;
6153                    CONT            & AM2901 YEXP,XEXP,RAMF,ADD,AB
6154    004C9   /                       & CARRYH        ; ADD EXPONENTS
6155                    CONT            & AM2901 XL,ZL,RAMF,PASS,ZA
6156    004CA   /                       & SPNOP         ; ZL = XL;
6157            ;
6158            ;       (R0=XU INITIALLY)
6159            ;       Q   := YU
6160            ;       ZU := 0      R.S. Q INTO Q0BUF   CNTR = SHIFT OP = 14
6161            ;       MULTIPLY; (ZU,Q) := XU * YU  (SEE CODE FOR 'MPY')
6162            ;
6163                    CONT            & AM2901 ,YU,QREG,PASS,ZB
6164    004CB   /                       & SPNOP         ;
6165                    PUSH  H#00E     & AM2901 ,ZU,SRAMQR,AND,ZB
6166            /                       & CARRYL        ;
6167    004CC   /                       & SHIFT B#1110  ;
6168                    RFCT            & AM2901 MPY,ZU,SRAMQR,ADD,AB
6169            /                       & CARRYL        ;
6170    004CD   /                       & SHIFT B#1110  ;
6171                    CONT            & AM2901 MPY,ZU,SRAMQR,SUBR,AB
6172            /                       & CARRYL        ;
6173    004CE   /                       & SHIFT B#1110  ;
6174                    CONT            & AM2901 ,XL,RAMF,PASS,ZQ
6175    004CF   /                       & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

6177            ;       YL := YL LS 4       (FOR FMPYSUB)
6178            ;       XL := XL + XU*YL; PROPAGATE CARRY.
6179            ;       YL := ZL LS 4       (DONE IN FMPYSUB)
6180            ;       XU  := YU
6181            ;       XL := XL + R0*YL; PROPAGATE CARRY.
6182            ;       XU := ZU
6183            ;
6184            ;
6185                    CALL  FMPYSUB    & AM2901 YL,YL,RAMA,PASS,DZ
6186    004D0   /                        & SPRD RL4        ;
6187                    CONT             & AM2901 ZL,YL,RAMA,PASS,DZ
6188    004D1   /                        & SPRD RL4        ;
6189                    CALL  FMPYSUB    & AM2901 YU,XU,RAMF,PASS,ZA
6190    004D2   /                        & SPNOP           ;
6191                    JP    NORM       & AM2901 ZU,XU,RAMF,PASS,ZA
6192    004D3   /                        & SPNOP           ;
```

```
LINE    ADDR    STATEMENT

6194            ;       MULTIPLY SUBROUTINE; MULTIPLIES XU*YL, AND ADDS THE
6195            ;       MSW TO (ZU,XL).  XU IS SIGNED, AND THE LOWER 8 BITS
6196            ;       OF YL ARE ZERO.
6197            ;       THE PRODUCT IS DEVELOPED IN (YL,Q); IF IT IS NEGATIVE,
6198            ;       WE MUST SUBTRACT ONE FROM ZU.  IF THE ADD TO XL HAS
6199            ;       A CARRY OUT, WE MUST ADD ONE TO ZU.
6200            ;
6201            ;       YL := YL LS 4  (COMPLETE SWAPPING BYTES)
6202            ;       Q   := YL
6203            ;       (DO 8 MULTIPLY STEPS; RESULT -> (YL,Q)
6204            ;
6205            FMPYSUB: CONT           & AM2901 YL,YL,RAMA,PASS,DZ
6206    004D4   /                       & SPRD RL4      ;
6207                     LDCT  H#007    & AM2901 ,YL,QREG,PASS,ZB
6208    004D5   /                       & LODMSR RESET  ; YL TO QREG
6209                     CPUSH          & AM2901 ,YL,SRAMQR,AND,ZB
6210            /                       & CARRYL        ;
6211            /                       & CONDMSR SGN   ;
6212    004D6   /                       & SHIFT B#1110  ; NEVER LOAD COUNTER
6213                     RFCT           & AM2901 MPY,YL,SRAMQR,ADD,AB
6214            /                       & CARRYL        ;
6215            /                       & SHIFT B#1110  ;
6216            /                       & LODMSR        ;
6217    004D7   /                       & ENBLO         ;
6218            ;
6219            ;       (BIT16 INDICATES IF PRODUCT < 0)
6220            ;       XL := XL + YL;  PROPAGATE CARRY TO ZU.
6221            ;       YL := ZL LS 4  (HAVE TO USE THE CYCLE ANYWAY)
6222            ;       IF PRODUCT<0, DECREMENT ZU.
6223            ;
6224                     CONT           & AM2901 YL,XL,RAMF,ADD,AB
6225            /                       & CARRYL        ;
6226    004D8   /                       & LODUSR        ;
6227                     CONT           & AM2901 ,ZU,RAMF,ADD,ZB
6228    004D9   /                       & CARRYUC       ;
6229                     RET            & AM2901 ,ZU,RAMF,SUBR,ZB
6230            /                       & CARRYEXT      ;
6231    004DA   /                       & CONDMSR SLT   ; SUB 1 IF LESS THAN
```

```
LINE    ADDR    STATEMENT

6233            ;*********************************************************************
6234            ;                                                                    *
6235            ;         '.FDV' FLOATING-POINT DIVIDE.                              *
6236            ;                                                                    *
6237            ;         DIVIDE THE UNPACKED NUMBERS                                *
6238            ;                                                                    *
6239            ;             (XU,XL,XEXP) / (YU,YL,YEXP) --> (XU,XL,ZEXP)           *
6240            ;                                                                    *
6241            ;         THE OPERANDS MUST BE NORMALIZED OR ZERO.                   *
6242            ;                                                                    *
6243            ;         OVERFLOW AND UNDERFLOW ARE POSSIBLE;  DIVIDE BY ZERO       *
6244            ;         IS TREATED AS OVERFLOW.  THE RESULT MAY BE                 *
6245            ;         UNNORMALIZED BY ONE BIT.                                   *
6246            ;                                                                    *
6247            ;*********************************************************************
6248            ;
6249            ;         READ SECOND OPERAND FROM MEMORY; UNPACK BOTH OPERANDS.
6250            ;
6251            .FDV:     CALL  DIARG       & AM2901 BR,TABQ,QREG,ADD,DZ
6252            /                           & CARRYL          ;
6253            /                           & LODUSR          ;
6254            /                           & LDMAR           ;
6255   004DB    /                           & DREAD           ; GET DIVISOR IN <Y>
6256                      CALL  UNPACK      & AM2901 XL,XEXP,RAMF,PASS,ZA
6257            /                           & LODMSR SWAPEO   ;
6258   004DC    /                           & ENBLC & ENBLO   ;
6259            ;
6260            ;         (S0,Q) = (XU,XL) RS 2
6261            ;         XEXP := XEXP - YEXP
6262            ;         IF S0<0 THEN
6263            ;           S0 := -S0
6264            ;           Q  := -Q  - BORROW
6265            ;         XU := XU .XOR. YU
6266            ;         XL := 0
6267            ;         IF YU<0 THEN
6268            ;           YL := -YL
6269            ;           YU := -YU - BORROW
6270            ;
6271                      CONT              & AM2901 ,XL,QREG,PASS,ZB
6272   004DD    /                           & SPNOP           ;
6273                      CONT              & AM2901 XU,S0,SRAMQR,ADD,ZA
6274            /                           & CARRYL          ;
6275   004DE    /                           & SHIFT B#1110    ;
6276                      CONT              & AM2901 YEXP,S0,SRAMQR,ADD,ZB
6277            /                           & CARRYL          ;
6278            /                           & LODUSR          ;
6279   004DF    /                           & SHIFT B#1110    ;
6280                      CJP   FDIV1       & AM2901 YEXP,XEXP,RAMF,SUBR,AB
6281            /                           & CARRYL          ;
6282   004E0    /                           & CONDUSR NSGN    ;
6283                      CONT              & AM2901 ,,QREG,SUBS,ZQ
6284            /                           & CARRYH          ;
6285   004E1    /                           & LODUSR          ;
```

```
LINE    ADDR    STATEMENT

6286                    CONT            & AM2901 YU,S0,RAMF,SUBS,ZB
6287    004E2   /                       & CARRYUC       ;
6288            FDIV1:  CONT            & AM2901 YU,YEXP,RAMF,PASS,ZA
6289    004E3   /                       & LODMSR        ;
6290                    CJP   FOFL      & AM2901 XU,YU,RAMF,EXOR,AB
6291    004E4   /                       & CONDMSR Z     ;
6292                    CJP   FDIV2     & AM2901 ,XL,RAMF,AND,ZB
6293            /                       & CARRYL        ; SO THERE IS NO OVERFLOW
6294    004E5   /                       & CONDMSR NSGN  ; LOADS USR
6295                    CONT            & AM2901 ,YL,RAMF,SUBS,ZB
6296            /                       & CARRYH        ;
6297    004E6   /                       & LODUSR        ;
6298                    LDCT  FDVPAT    & AM2901 ,YEXP,RAMF,SUBS,ZB
6299    004E7   /                       & CARRYUC       ;
6300            ;
6301            ;       YEXP := YEXP + 1
6302            ;       CALL FDIVSUB TO DIVIDE:
6303            ;         S0 := REM (W/O LAST CORRECTION)
6304            ;         Q  := .NOT. QUOTIENT
6305            ;
6306            FDIV2:  JSRP  FDIVSUB   & AM2901 ,YEXP,RAMF,ADD,ZB
6307            /                       & CARRYH        ;
6308    004E8   /                       & CONDUSR NOVR  ;
```

```
LINE    ADDR    STATEMENT

6310            ;       IF S0<0 THEN
6311            ;         S0 := S0 + YEXP
6312            ;       ZU := .NOT. Q
6313            ;       IF ZU<0 THEN RESULT := 0
6314            ;       S0 := S0 + ZU
6315            ;
6316                    CONT            & AM2901 ,ZU,RAMF,EXNOR,ZQ
6317    004E9   /                       & LODUSR        ; ZU =.NOT. Q
6318                    CJP   FDIV5     & AM2901 ZU,S0,NOP,PASS,ZB
6319    004EA   /                       & CONDUSR SGN   ; IF <0, RES=0
6320                    CJP   FDIV3     & AM2901 ZU,S0,RAMF,ADD,AB
6321            /                       & CARRYL        ;
6322    004EB   /                       & CONDUSR NSGN  ;
6323                    CONT            & AM2901 YEXP,S0,RAMF,ADD,AB
6324    004EC   /                       & CARRYL        ; S0= S0 + YEXP
6325            ;
6326            ;       A := ZU
6327            ;       YL := YL LS 4
6328            ;       CALL FMPYSUB TO:  XL := YL*A (UPPER)
6329            ;
6330            FDIV3:  CONT            & AM2901 ZU,A,RAMF,PASS,ZA
6331    004ED   /                       & SPNOP         ;
6332                    CALL  FMPYSUB   & AM2901 YL,YL,RAMA,PASS,DZ
6333    004EE   /                       & SPRD RL4      ;
6334            ;
6335            ;       S0 := S0-XL
6336            ;       (S0,Q) := (S0,Q) RS 2
6337            ;       XEXP := XEXP + 1
6338            ;       CALL FDIVSUB DIVIDE:  Q := .NOT. QUOTIENT
6339            ;
6340                    CONT            & AM2901 XL,S0,SRAMQR,SUBR,AB
6341            /                       & CARRYL        ;
6342    004EF   /                       & SHIFT B#0110  ;
6343                    CONT            & AM2901 ,S0,SRAMQR,PASS,ZB
6344            /                       & SHIFT B#0110  ;
6345    004F0   /                       & SPNOP         ;
6346                    CALL  FDIVSUB   & AM2901 ,XEXP,RAMF,ADD,ZB
6347    004F1   /                       & CARRYH        ;
```

```
LINE    ADDR    STATEMENT

6349            ;       XL := .NOT. Q
6350            ;       Q := 0
6351            ;       (XL,Q) := (XL,Q) LS 2 CIRCULAR
6352            ;
6353                    CONT            & AM2901 ,XL,RAMF,EXNOR,ZQ
6354    004F2   /                       & SPNOP         ;
6355                    PUSH  H#001     & AM2901 ,,QREG,AND,ZQ
6356    004F3   /                       & SPNOP         ;
6357                    RFCT            & AM2901 ,XL,SRAMQL,PASS,ZB
6358    004F4   /                       & SHIFT B#1111  ;
6359            ;
6360            ;       XU := ZU + Q
6361            ;       IF OPERAND SIGNS DIFFER, NEGATE RESULT.
6362            ;
6363                    CONT            & AM2901 ZU,YU,NOP,PASS,ZB
6364    004F5   /                       & LODUSR        ;
6365                    CJP   NORM      & AM2901 ZU,XU,RAMF,ADD,AQ
6366            /                       & CARRYL        ;
6367    004F6   /                       & CONDUSR NSGN  ;
6368                    CONT            & AM2901 ,XL,RAMF,SUBS,ZB
6369            /                       & CARRYH        ;
6370    004F7   /                       & LODUSR        ;
6371                    JP    NORM      & AM2901 ,XU,RAMF,SUBS,ZB
6372    004F8   /                       & CARRYUC       ;
6373            ;
6374            ;       ZERO DIVIDEND (FOUND OUT THE HARD WAY).
6375            ;       XU := 0    (XL=0 ALREADY)
6376            ;
6377            FDIV5:  JP    NORM      & AM2901 ,XU,RAMF,AND,ZB
6378    004F9   /                       & SPNOP         ;
6379            ;
6380            ;       DIVIDE SUBROUTINE; DIVIDES (S0,Q) BY YEXP.
6381            ;       THE FINAL REMAINDER RESTORE IS NOT DONE,
6382            ;       SINCE ON THE SECOND CALL THE REMAINDER IS DISCARDED.
6383            ;
6384            FDIVSUB: PUSH H#010     & AM2901 YEXP,S0,NOP,PASS,ZB
6385    004FA   /                       & DIVUCOND      ;
6386                    RFCT            & AM2901 YEXP,DIV,SRAMQL,SUBR,AB
6387            /                       & CARRYL        ;
6388            /                       & DIVUCOND      ;
6389    004FB   /                       & SHIFT B#1111  ;
6390                    RET             & AM2901 ,S0,SRAMR,PASS,ZB
6391    004FC   /                       & SHIFT B#1111  ;
```

```
LINE    ADDR    STATEMENT

6393            ;*****************************************************
6394            ;                                                    *
6395            ;         UNPACK.  FETCH 2ND OPERAND & UNPACK BOTH:  *
6396            ;                                                    *
6397            ;            (A,B) => (XU,XL,XEXP)                   *
6398            ;                                                    *
6399            ;            (MEM,MEM+1) => (YU,YL,YEXP)             *
6400            ;                                                    *
6401            ;         NOTE THAT A=XU AND B=XL.                   *
6402            ;         TO UNPACK (A,B) ONLY (FOR FIX), ENTER AT   *
6403            ;         'UNPACK1' WITH E & 0 SWAPPED AND XEXP=B.   *
6404            ;                                                    *
6405            ;*****************************************************
6406            ;
6407            ;
6408            ;         YL   := (YEXP .AND. FF00)
6409            ;         YEXP := (YEXP .XOR. YL) SHIFT RIGHT INTO MC
6410            ;         IF YEXP<0 THEN
6411            ;           YEXP := YEXP .OR. FF80
6412            ;         EXIT
6413            ;
6414            UNPACK:   CONT             & AM2901 YEXP,YL,RAMF,AND,DA
6415   004FD    /                          & IMM H#FF00      ;
6416                      CONT             & AM2901 YL,YEXP,SRAMR,EXOR,AB
6417   004FE    /                          & SHIFT B#0010    ;
6418                      CJP   UNPACK1    & AM2901 YEXP,YEXP,NOP,ADD,AQ
6419   004FF    /                          & CONDMSR NC      ; DOES NOTHING
6420                      CONT             & AM2901 YEXP,YEXP,RAMF,OR,DA
6421   00500    /                          & IMM H#FF80      ;
6422            ;
6423            ;         XL   := (XEXP .AND. FF00)
6424            ;         XEXP := (XEXP .XOR. XL) RIGHT SHIFT INTO MC
6425            ;         IF MC=1 THEN
6426            ;           XEXP := XEXP .OR. FF80
6427            ;         MC := 0
6428            ;         SWAP E,0 BACK:  E RESTORED, 0=0.
6429            ;
6430            UNPACK1:  CONT             & AM2901 XL,XL,RAMF,AND,DA
6431   00501    /                          & IMM H#FF00      ; MASK OFF EXPONENT
6432                      CONT             & AM2901 XL,XEXP,SRAMR,EXOR,AB
6433   00502    /                          & SHIFT B#0010    ; YEXP = EXP, MC=EXP SIGN
6434                      CJP   UNPACK2    & AM2901 ,,NOP,ADD,AQ
6435   00503    /                          & CONDMSR NC      ; IF NOT NEG EXP, JP
6436                      CONT             & AM2901 XEXP,XEXP,RAMF,OR,DA
6437   00504    /                          & IMM H#FF80      ; SIGN EXTEND EXPONENT
6438                      CONT             & AM2901 ,,NOP,ADD,AQ
6439            /                          & LODMSR RESET    ;
6440   00505    /                          & ENBLC           ; CLEAR MC FOR SWAP
6441            UNPACK2:  RET              & AM2901 ,XEXP,NOP,PASS,ZB
6442            /                          & LODMSR SWAPEO   ;
6443   00506    /                          & ENBLC & ENBLO   ;
```

```
LINE    ADDR    STATEMENT

6445            ;*************************************************************************
6446            ;*                                                                       *
6447            ;*        Double Precision Floating Microcode                            *
6448            ;*                                                                       *
6449            ;*   This module includes:                                               *
6450            ;*                                                                       *
6451            ;*     - Double Precision Floating Point instructions                    *
6452            ;*                                                                       *
6453            ;*         .TADD                                                         *
6454            ;*         .TSUB                                                         *
6455            ;*         .TMPY                                                         *
6456            ;*         .TDIV                                                         *
6457            ;*         .TFTS                                                         *
6458            ;*         .TFTD                                                         *
6459            ;*         .TFXS                                                         *
6460            ;*         .TFXD                                                         *
6461            ;*                                                                       *
6462            ;*************************************************************************
```

```
LINE    ADDR    STATEMENT

6464            ;*********************************************************************
6465            ;*    Lightning Double Precision Floating Point Microcode            *
6466            ;*        Version 1.0  B.G. 08/24/82                                 *
6467            ;*        MODIFIED APRIL 1983 FOR THUNDER PROCESSOR   JFM            *
6468            ;*                                                                   *
6469            ;* A,B,X,Y,Q,Z,PC preserved.        MP: result sign and exp.        *
6470            ;* S0-S3: mantissa 1st op, result.  S4-S7: mantissa 2nd opnd.        *
6471            ;*                                                                   *
6472            ;*********************************************************************
6473            ;
6474            ;         FETCH DOUBLE PRECISION ARGUMENTS.
6475            ;
6476            ;         At DBLARG1: Read of first word must have been started.
6477            ;                     argument is read & unpacked into S0-S3
6478            ;                     and MP; Q is destroyed.
6479            ;         At DBLARG2: As above, but into S4-S7 and Q.
6480            ;         DBLARG1 copies (E) into (O) and clobbers (E).
6481            ;         DBLARG2 clobbers (E) but not (O).
6482            ;
6483            DBLARG1:  CALL  READRES    & AM2901 ,,NOP,ADD,AQ
6484   00507    /                          & SPNOP          ;
6485                      CONT             & AM2901 ,,QREG,ADD,ZQ
6486            /                          & CARRYH         ;
6487            /                          & LODMSR SWAPEO  ;
6488            /                          & ENBLC & ENBLO  ;
6489   00508    /                          & LDMAR          ;
6490            DBLARG1A: CALL  INCQ       & AM2901 TAB,S0,RAMF,PASS,DZ
6491   00509    /                          & DREAD          ; S0 = 1ST WORD, READ 2ND W
6492                      CALL  INCQ       & AM2901 TAB,S1,RAMF,PASS,DZ
6493   0050A    /                          & DREAD          ; S1 = 2ND WORD, READ 3RD W
6494                      CONT             & AM2901 TAB,S2,RAMF,PASS,DZ
6495   0050B    /                          & DREAD          ; S2 = 3RD WORD, READ 4TH W
6496                      CONT             & AM2901 ,S3,RAMF,EXNOR,DZ
6497   0050C    /                          & IMM H#FF00     ; S3 = EXP MASK
6498                      CONT             & AM2901 TAB,,QREG,PASS,DZ
6499   0050D    /                          & SPNOP          ; Q = 4TH WORD
6500                      CONT             & AM2901 S3,MP,SRAMR,AND,AQ
6501   0050E    /                          & SHIFT B#0010   ; MP = EXP ; MC = EXP SIGN
6502                      CRET             & AM2901 S3,S3,RAMF,NOTRS,AQ
6503   0050F    /                          & CONDMSR NC     ; S3 = REMAINDER OF MANTISS
6504            ;                                            RETURN IF EXP IS POSITIVE
6505                      RET              & AM2901 MP,MP,RAMF,OR,DA
6506   00510    /                          & IMM H#FF80     ; SIGN EXTEND EXPONENT, RET
```

```
LINE    ADDR    STATEMENT

6508            DBLARG2:  CALL  READRES    & AM2901 ,,NOP,ADD,AQ
6509            /                          & LODMSR SWAPEO   ; SWAP E & O BACK
6510    00511   /                          & ENBLC & ENBLO   ; IN CASE READIND GETS INT'
6511                      CONT             & AM2901 ,,QREG,ADD,ZQ
6512            /                          & CARRYH          ;
6513            /                          & LODMSR SWAPEO   ;
6514            /                          & ENBLC & ENBLO   ; E & O SWAPPED AGAIN
6515    00512   /                          & LDMAR           ; MAR = 2ND WORD ADDRESS
6516            DBLARG2A: CALL  INCQ       & AM2901 TAB,S4,RAMF,PASS,DZ
6517    00513   /                          & DREAD           ; S4 = 1ST WORD, READ 2ND W
6518                      CALL  INCQ       & AM2901 TAB,S5,RAMF,PASS,DZ
6519    00514   /                          & DREAD           ; S5 = 2ND WORD, READ 3RD W
6520                      CONT             & AM2901 TAB,S6,RAMF,PASS,DZ
6521    00515   /                          & DREAD           ; S6 = 3RD WORD, READ 4TH W
6522                      CONT             & AM2901 ,S7,QREG,PASS,DZ
6523    00516   /                          & IMM H#007F      ; QREG = EXP MASK
6524                      CONT             & AM2901 TAB,S7,SRAMR,PASS,DZ
6525    00517   /                          & SHIFT B#1000    ; S7 = 4TH WORD SR1, MC=EXP
6526                      CJP   DBLARG2B   & AM2901 S7,S7,QREG,AND,AQ
6527    00518   /                          & CONDMSR NC      ; JP IF EXPONENT POSITIVE
6528                      CONT             & AM2901 S7,S7,QREG,OR,DQ
6529    00519   /                          & IMM H#FF80      ; SIGN EXTEND EXPONENT
6530            DBLARG2B: RET              & AM2901 S7,S7,SRAML,AND,DA
6531    0051A   /                          & IMM H#FF80      ; S7 = REMAINDER; SHIFT IS
6532            ;                                            ; MC IS SIGN OF EXPONENT
```

```
LINE    ADDR    STATEMENT

6535            ;*                                                              *
6536            ;*      Fix double precision to single integer.                 *
6537            ;*      Result ends up in A register                            *
6538            ;*                                                              *
6539            ;*      JSB .TFXS                                               *
6540            ;*      DEF OPERAND                                             *
6541            ;*                                                              *
6542            ;****************************************************************
6543            ;       First, check for negative exponent and
6544            ;       exponent >= 15.
6545            ;
6546            .TFXS:  CCALL READRES   & AM2901 BR,TABQ,QREG,ADD,DZ
6547            /                       & CARRYL        ;
6548            /                       & CONDEXT MDIR15 ;
6549            /                       & LDMAR         ;
6550   0051B    /                       & DREAD         ;
6551                    CALL  DBLARG2A  & AM2901 ,,QREG,ADD,ZQ
6552            /                       & CARRYH        ;
6553            /                       & LODMSR SWAPEO ;
6554            /                       & ENBLC & ENBLO ;
6555   0051C    /                       & LDMAR         ;
6556                    CONT            & AM2901 PC,MP,RAMA,SUBS,DQ
6557            /                       & CARRYH        ;
6558            /                       & LUSRCOND      ;
6559            /                       & IMM H#000F    ;
6560   0051D    /                       & LDMAR         ; MAR = NEXTINST
6561                    CJP   TFXS3     & AM2901 S7,S5,RAMF,OR,AB
6562   0051E    /                       & CONDUSR SLE   ;
6563                    CJP   TFXS1     & AM2901 S6,S5,QREG,AND,ZQ
6564   0051F    /                       & CONDMSR NC    ; MSR C SET IN DBLARG2
6565                    CONT            & AM2901 ,A,RAMF,AND,ZB
6566            /                       & LODMSR SWAPEO ;
6567            /                       & ENBLC & ENBLO ;
6568   00520    /                       & IFETCH        ;
6569            TEND:   JZ              & AM2901 ,PC,RAMF,ADD,ZB
6570            /                       & CARRYH        ;
6571            /                       & LODMSR RESET  ;
6572   00521    /                       & ENBLO         ; CLEAR 0
6573            ;
6574            ;       Load the counter for (15-exp) loops; shift (A,Q).
6575            ;       Also start the next instruction fetch.
6576            ;
6577            TFXS1:  CONT            & AM2901 S6,S5,RAMF,OR,AB
6578            /                       & LODMSR SWAPEO ;
6579   00522    /                       & ENBLC & ENBLO ;
6580                    CONT            & AM2901 S4,A,RAMF,PASS,ZA
6581   00523    /                       & SPNOP         ;
6582                    PUSH            & AM2901 ,MP,RAMF,SUBR,ZB
6583            /                       & CARRYH        ;
6584   00524    /                       & SPETC LDCTY   ;
6585                    RFCT            & AM2901 S5,A,SRAMQR,ADD,ZB
6586            /                       & CARRYL        ;
```

```
LINE    ADDR    STATEMENT

6587            /                           & SHIFT B#1110    ;
6588    00525   /                           & LODUSR          ;
6589            ;
6590            ;       If value negative and bits shifted off, add one.
6591            ;
6592            TFXS2:  CJP   TEND          & AM2901 S5,PC,NOP,OR,AQ
6593            /                           & CONDUSR NSGN    ;
6594    00526   /                           & CMGO & IFETCH   ;
6595                    JP    TEND          & AM2901 ,A,RAMF,ADD,ZB
6596            /                           & CARRYEXT        ;
6597            /                           & CONDUSR NZ      ;
6598    00527   /                           & IFETCH          ;
6599            ;
6600            ;       Exponent is > 14; 15 => no shift, other => ofl.
6601            ;
6602            TFXS3:  CONT                & AM2901 ,S6,QREG,PASS,ZB
6603            /                           & LODMSR SWAPEO   ;
6604    00528   /                           & ENBLC & ENBLO   ;
6605                    CONT                & AM2901 ,MP,NOP,PASS,ZB
6606    00529   /                           & LODUSR          ;
6607                    CJP   TFXS2         & AM2901 S4,A,RAMF,PASS,ZA
6608    0052A   /                           & CONDUSR Z       ;
6609            TFXSOV: CONT                & AM2901 ,A,RAMF,EXNOR,DZ
6610            /                           & IMM H#8000      ;
6611    0052B   /                           & IFETCH          ;
6612                    JZ                  & AM2901 ,PC,RAMF,ADD,ZB
6613            /                           & CARRYH          ;
6614            /                           & LODMSR SET      ;
6615    0052C   /                           & ENBLO           ;
```

```
LINE    ADDR    STATEMENT

6617            ;*********************************************************************
6618            ;*                                                                   *
6619            ;*       Float single integer to double precision.                   *
6620            ;*                                                                   *
6621            ;*       JSB .TFTS                                                   *
6622            ;*       DEF RESULT                                                  *
6623            ;*                                                                   *
6624            ;*********************************************************************
6625            ;
6626            ;       First, test for zero & clear 0.
6627            ;
6628            .TFTS:  CONT            & AM2901 ,S2,RAMF,AND,ZB
6629            /                       & LODMSR SWAPEO  ; S2 = 0
6630   0052D    /                       & ENBLC & ENBLO  ; SWAP E & 0
6631                    LDCT  DNCOUNT1  & AM2901 A,S0,RAMF,PASS,ZA
6632   0052E    /                       & LODUSR         ; S0 = A
6633            TFTS1:  CJP   TFTS2     & AM2901 ,S1,RAMF,AND,ZB
6634   0052F    /                       & CONDUSR NZ     ; S1 = 0
6635                    JP    TFTS3     & AM2901 PC,S3,RAMF,AND,ZB
6636   00530    /                       & SPNOP          ; S3 = 0
6637            ;
6638            ;       Normalize:  shift until the most significant
6639            ;       bit is shifted out.  Then shift msb and sign
6640            ;       bit back in from E and .NOT. E
6641            ;
6642            TFTS2:  CPUSH           & AM2901 ,S0,SRAML,PASS,ZB
6643            /                       & CONDMSR SGN    ; NEVER LOAD COUNTER
6644   00531    /                       & SHIFT B#0000   ; IRSKP = S0
6645                    TWB             & AM2901 ,S0,SRAML,PASS,ZB
6646            /                       & CONDEXT IRSKBF ;
6647   00532    /                       & SHIFT B#0000   ;
6648                    CONT            & AM2901 ,S0,SRAMR,PASS,ZB
6649            /                       & LODMSR INVERT  ;
6650            /                       & ENBLC          ; INVERT MC
6651   00533    /                       & SHIFT B#0100   ; RESTORE MSB
6652                    JSRP            & AM2901 ,S0,SRAMR,PASS,ZB
6653            /                       & CONDMSR NSGN   ; ALWAYS JSR
6654   00534    /                       & SHIFT B#0100   ; RESTORE SIGN
6655                    CONT            & AM2901 S7,S3,SRAML,SUBS,DA
6656            /                       & CARRYH         ;
6657   00535    /                       & IMM H#0010     ; Shift = 0000.
6658            ;
6659            ;       Write result & exit.
6660            ;
6661            TFTS3:  JP    DSTORE    & AM2901 PC,S5,RAMF,PASS,ZA
6662            /                       & LODMSR RESET   ; PC POINTS TO NEXT INST
6663   00536    /                       & ENBLC          ; CLEAR MC (=M0 AFTER DSTOR
```

```
LINE    ADDR    STATEMENT

6665            ;*********************************************************************
6666            ;*                                                                   *
6667            ;*       Fix double precision to double integer.                     *
6668            ;*       Double integer ends up in A & B                             *
6669            ;*                                                                   *
6670            ;*       JSB .TFXD                                                   *
6671            ;*       DEF OPERAND                                                 *
6672            ;*                                                                   *
6673            ;*********************************************************************
6674            ;       Get value; test for exponent > 14.
6675            ;
6676            .TFXD:  CCALL READRES   & AM2901 BR,TABQ,QREG,ADD,DZ
6677            /                       & CARRYL        ;
6678            /                       & CONDEXT MDIR15 ;
6679            /                       & LDMAR         ;
6680    00537   /                       & DREAD         ;
6681                    CALL  DBLARG2A  & AM2901 ,,QREG,ADD,ZQ
6682            /                       & CARRYH        ;
6683            /                       & LODMSR SWAPEO  ;
6684            /                       & ENBLC & ENBLO  ;
6685    00538   /                       & LDMAR         ;
6686                    CONT            & AM2901 PC,MP,RAMA,SUBS,DQ
6687            /                       & CARRYH        ;
6688            /                       & IMM H#000F    ;
6689            /                       & LUSRCOND      ;
6690    00539   /                       & LDMAR         ; MAR = NEXTINST
6691                    CJP   TFXD4     & AM2901 S7,S6,RAMF,OR,AB
6692    0053A   /                       & CONDUSR SLE   ;
6693            ;
6694            ;       Exponent small.  If < 0, return 0; else shift.
6695            ;
6696                    CJP   TFXD1     & AM2901 S6,S5,RAMF,OR,AB
6697    0053B   /                       & CONDMSR NC    ; MSR C = EXP SIGN FRM DBLA
6698                    CONT            & AM2901 ,A,RAMF,AND,ZQ
6699            /                       & LODMSR SWAPEO  ;
6700    0053C   /                       & ENBLC & ENBLO  ;
6701                    JP    TEND      & AM2901 ,B,RAMF,AND,ZQ
6702    0053D   /                       & IFETCH        ;
6703            TFXD1:  CONT            & AM2901 ,,QREG,AND,ZQ
6704            /                       & LODMSR SWAPEO  ;
6705    0053E   /                       & ENBLC & ENBLO  ;
6706                    CONT            & AM2901 S4,B,RAMF,PASS,ZA
6707    0053F   /                       & SPNOP         ;
6708                    PUSH            & AM2901 ,MP,NOP,SUBR,ZB
6709            /                       & CARRYH        ;
6710    00540   /                       & SPETC LDCTY   ;
6711                    RFCT            & AM2901 S5,B,SRAMQR,ADD,ZB
6712            /                       & CARRYL        ;
6713            /                       & SHIFT B#1110  ;
6714    00541   /                       & LODUSR        ;
```

```
LINE    ADDR    STATEMENT

6716            ;       If negative & bits lost, increment it.  Extend sign.
6717            ;
6718                    CJP   TFXD3     & AM2901 S5,B,NOP,OR,AQ
6719    00542   /                       & CONDUSR NSGN   ;
6720                    CONT            & AM2901 ,B,RAMF,ADD,ZB
6721            /                       & CARRYEXT       ;
6722    00543   /                       & CONDUSR NZ     ; IF NEG AND BITS, ADD 1
6723            TFXD2:  JP    TEND      & AM2901 A,A,RAMF,SUBR,AB
6724            /                       & CARRYEXT       ;
6725            /                       & CONDUSR NC     ;
6726    00544   /                       & IFETCH         ;
6727            TFXD3:  JP    TEND      & AM2901 A,A,RAMF,EXOR,AB
6728    00545   /                       & IFETCH         ;
6729            ;
6730            ;       Exponent > 14; test for exp >= 31, take abs value.
6731            ;
6732            TFXD4:  CONT            & AM2901 MP,MP,RAMF,ADD,DA
6733            /                       & CARRYL         ;
6734            /                       & IMM H#0010     ;
6735    00546   /                       & LUSRCOND       ;
6736                    CJP   TFXD7     & AM2901 S4,A,RAMF,PASS,ZA
6737            /                       & CARRYL         ;
6738    00547   /                       & CONDUSR SLE    ;
6739                    CJP   TFXD5     & AM2901 ,S5,QREG,PASS,ZB
6740    00548   /                       & CONDUSR NSGN   ;
6741                    CONT            & AM2901 ,S6,NOP,SUBS,ZB
6742            /                       & CARRYH         ;
6743    00549   /                       & LODUSR         ;
6744                    CONT            & AM2901 ,,QREG,SUBS,ZQ
6745    0054A   /                       & CARRYUC        ;
6746                    CONT            & AM2901 ,A,RAMF,SUBS,ZB
6747    0054B   /                       & CARRYUC        ;
```

```
LINE    ADDR    STATEMENT

6749            ;       Shift.  If original value was negative, negate.
6750            ;
6751            TFXD5:  PUSH            & AM2901 ,MP,RAMF,SUBR,ZB
6752            /                       & CARRYH         ;
6753    0054C   /                       & SPETC LDCTY    ;
6754                    RFCT            & AM2901 ,A,SRAMQR,PASS,ZB
6755    0054D   /                       & SHIFT B#0110   ;
6756                    CONT            & AM2901 ,S4,NOP,PASS,ZB
6757            /                       & LODMSR SWAPEO  ;
6758    0054E   /                       & ENBLC & ENBLO  ;
6759                    CJP   TEND      & AM2901 ,B,RAMF,PASS,ZQ
6760            /                       & CONDMSR NSGN   ;
6761    0054F   /                       & CMGO & IFETCH  ;
6762                    CONT            & AM2901 ,B,RAMF,SUBS,ZB
6763            /                       & CARRYH         ;
6764    00550   /                       & LODUSR         ;
6765                    JP    TEND      & AM2901 ,A,RAMF,SUBS,ZB
6766            /                       & CARRYUC        ;
6767    00551   /                       & IFETCH         ;
6768            ;
6769            ;       Exp >= 31; overflow or no shift.
6770            ;
6771            TFXD7:  CONT            & AM2901 S5,MP,NOP,PASS,ZB
6772            /                       & LODMSR SWAPEO  ;
6773    00552   /                       & ENBLC & ENBLO  ;
6774                    CJP   TFXD8     & AM2901 S5,B,RAMF,PASS,ZA
6775    00553   /                       & CONDUSR NZ     ;
6776                    CONT            & AM2901 ,S4,NOP,PASS,ZB
6777    00554   /                       & LODUSR         ;
6778                    CJP   TEND      & AM2901 ,S6,NOP,PASS,ZB
6779            /                       & CONDUSR NSGN   ;
6780    00555   /                       & CMGO & IFETCH  ;
6781                    CONT            & AM2901 ,B,RAMF,ADD,ZB
6782            /                       & CARRYEXT       ;
6783    00556   /                       & CONDUSR NZ     ; IF S6<>0, INC B
6784                    JP    TEND      & AM2901 ,A,RAMF,ADD,ZB
6785            /                       & CARRYUC        ;
6786    00557   /                       & IFETCH         ; IF CARRY FROM B, INC A
6787            ;
6788            TFXD8:  JP    TFXSOV    & AM2901 B,B,RAMF,EXNOR,AB
6789    00558   /                       & SPNOP          ; B = ONES
```

```
LINE    ADDR    STATEMENT

6791            ;*********************************************************************
6792            ;*                                                                   *
6793            ;*       Float double integer to double precision.                   *
6794            ;*       Double integer is in A & B                                  *
6795            ;*                                                                   *
6796            ;*       JSB .TFTD                                                   *
6797            ;*       DEF RESULT                                                  *
6798            ;*                                                                   *
6799            ;*********************************************************************
6800            .TFTD:  CONT            & AM2901 ,S2,RAMF,AND,ZB
6801            /                       & LODMSR SWAPEO  ; S2 = 0
6802   00559    /                       & ENBLC & ENBLO  ; SWAP E & O
6803                    LDCT  DNCOUNT1  & AM2901 A,S0,RAMF,OR,ZA
6804            /                       & CARRYH         ; SET COUNTER
6805   0055A    /                       & LODUSR         ; S0 = A
6806                    CJP   TFTD1     & AM2901 ,B,QREG,PASS,ZB
6807   0055B    /                       & CONDUSR UGT    ; IF NOT ZERO OR ONES
6808            ;
6809            ;       A is 0 or -1; if same as sign of B, use .TFTS
6810            ;
6811                    CONT            & AM2901 A,B,NOP,EXOR,AB
6812   0055C    /                       & LODUSR         ; SET SIGN
6813                    CJP   TFTS1     & AM2901 B,S0,RAMF,PASS,ZA
6814   0055D    /                       & CONDUSR NSGN   ; IF SIGNS THE SAME
6815                    CONT            & AM2901 A,S0,RAMF,PASS,ZA
6816   0055E    /                       & SPNOP          ;
6817            ;
6818            ;       A has significant bit; normalize.
6819            ;
6820            TFTD1:  CPUSH           & AM2901 ,S0,SRAMQL,PASS,ZB
6821            /                       & CONDMSR SGN    ; NEVER LOAD COUNTER
6822   0055F    /                       & SHIFT B#0100   ; IRSKP = 0
6823                    TWB             & AM2901 ,S0,SRAMQL,PASS,ZB
6824            /                       & CONDEXT IRSKBF ;
6825   00560    /                       & SHIFT B#0100   ; NORMALIZE
6826                    CONT            & AM2901 ,S0,SRAMQR,PASS,ZB
6827            /                       & LODMSR INVERT  ; E = SIGN
6828            /                       & ENBLC          ;
6829   00561    /                       & SHIFT B#0100   ; RESTORE MSB
6830                    JSRP            & AM2901 ,S0,SRAMQR,ADD,ZB
6831            /                       & CONDMSR NSGN   ; ALWAYS JSR
6832   00562    /                       & SHIFT B#0100   ; RESTORE SIGN
6833                    CONT            & AM2901 S7,S3,SRAML,SUBS,DA
6834            /                       & CARRYH         ;
6835   00563    /                       & IMM H#0020     ; Shift = 0000.
6836                    JP    TFTS3     & AM2901 ,S1,RAMF,PASS,ZQ
6837   00564    /                       & SPNOP          ; S1 = QREG
```

```
LINE    ADDR    STATEMENT

6839            ;*************************************************************************
6840            ;*                                                                       *
6841            ;*       Double precision subtract (.TSUB);                              *
6842            ;*                                                                       *
6843            ;*       JSB .TSUB                                                       *
6844            ;*       DEF DIFFERENCE                                                  *
6845            ;*       DEF MINUEND                                                     *
6846            ;*       DEF SUBTRAHEND                                                  *
6847            ;*                                                                       *
6848            ;*************************************************************************
6849            ;
6850            ;       Load 2nd operand into S4-S7, 1st operand into S0-S3
6851            ;       and negate S4-S7.
6852            ;
6853            .TSUB:  CALL  DBLARG1   & AM2901 ,PC,NOP,ADD,ZB
6854            /                       & CARRYH        ;
6855            /                       & LDMAR         ;
6856   00565    /                       & CREAD         ; GET ARG2 INTO (S0,S1,S2,S
6857                    CALL  DBLARG2   & AM2901 ,PC,NOP,PASS,ZB
6858            /                       & LDMAR         ;
6859   00566    /                       & CREAD         ; GET ARG1 INTO (S4,S5,S6,S
6860                    CALL  DNEGATE   & AM2901 ,S3,RAMF,SUBS,ZB
6861            /                       & CARRYH        ;
6862   00567    /                       & LODUSR        ; NEGATE FIRST ARG
6863                    JP    TADD0     & AM2901 ,S4,NOP,ADD,ZB
6864            /                       & CARRYL        ; JUMP TO ADD CODE
6865            /                       & LODMSR        ;
6866   00568    /                       & ENBLC         ;
6867            ;
6868            ;       Routine to negate (S0,S1,S2,S3) with exp in MP.
6869            ;       ENTER DNEGATE WITH S3 ALREADY NEGATED AND uC SET APPROPRIATEL
6870            ;
6871            DNEGATE: CONT           & AM2901 ,S2,RAMF,SUBS,ZB
6872   00569    /                       & CARRYUC       ;
6873                    CONT            & AM2901 ,S1,RAMF,SUBS,ZB
6874   0056A    /                       & CARRYUC       ;
6875                    CONT            & AM2901 ,S0,RAMF,SUBS,ZB
6876   0056B    /                       & CARRYUC       ;
6877                    CJP   DNEGATE1  & AM2901 S0,S0,NOP,ADD,AB
6878            /                       & CARRYL        ;
6879   0056C    /                       & CONDUSR NOVR  ;
6880                    CONT            & AM2901 ,S0,SRAMR,PASS,ZB
6881   0056D    /                       & SHIFT B#0000  ;
6882                    RET             & AM2901 ,MP,RAMF,ADD,ZB
6883   0056E    /                       & CARRYH        ;
6884            DNEGATE1: CRET          & AM2901 ,S0,NOP,PASS,ZB
6885   0056F    /                       & CONDUSR OVR   ;
6886                    CRET            & AM2901 ,S0,SRAML,PASS,ZB
6887            /                       & SHIFT B#0000  ;
6888   00570    /                       & CONDUSR Z     ; Return if zero.
6889                    RET             & AM2901 ,MP,RAMF,SUBR,ZB
6890   00571    /                       & CARRYH        ;
```

```
LINE    ADDR    STATEMENT

6892            ;*********************************************************************
6893            ;*                                                                   *
6894            ;*       Double precision add (.TADD)                                *
6895            ;*                                                                   *
6896            ;*       DEF .TADD                                                   *
6897            ;*       DEF SUM                                                     *
6898            ;*       DEF AUGEND                                                  *
6899            ;*       DEF ADDEND                                                  *
6900            ;*                                                                   *
6901            ;*********************************************************************
6902            ;
6903            ;       First, get the operands.
6904            ;
6905            .TADD:  CALL  DBLARG1   & AM2901 ,PC,NOP,PASS,ZB
6906            /                       & LDMAR         ;
6907   00572    /                       & CREAD         ; GET ARG1 IN (S0,S1,S2,S3)
6908                    CALL  DBLARG2   & AM2901 ,PC,NOP,ADD,ZB
6909            /                       & CARRYH        ;
6910            /                       & LDMAR         ;
6911   00573    /                       & CREAD         ; GET ARG2 IN (S4,S5,S6,S7)
6912            ;
6913            ;       Check for either operand zero;  Q := R12 - Q
6914            ;       Clear Mc and leave clear until much later.
6915            ;
6916                    CONT            & AM2901 ,S4,NOP,ADD,ZB
6917            /                       & CARRYL        ;
6918            /                       & LODMSR        ; CLEAR MC
6919   00574    /                       & ENBLC         ; MZ=ARG2
6920            TADD0:  CJP   DPACK     & AM2901 PC,S0,NOP,PASS,ZB
6921   00575    /                       & CONDMSR Z     ; DONE IF ARG2=0;uZ=ARG2
6922                    CJP   TADD1     & AM2901 MP,,QREG,SUBS,AQ
6923            /                       & CARRYH        ;  Q=EXPONENT DIFFERENCE
6924   00576    /                       & CONDUSR NZ    ; IF ARG2<>0,ADD
6925                    CALL  SWPDBL    & AM2901 MP,MP,RAMF,SUBS,AQ
6926   00577    /                       & CARRYH        ;
6927                    JP    DPACK     & AM2901 PC,S5,NOP,ADD,AQ
6928   00578    /                       & SPNOP         ;
6929            ;
6930            ;       If second exponent is larger, swap the operands.
6931            ;
6932            TADD1:  CJP   TADD2     & AM2901 ,,NOP,ADD,AQ
6933   00579    /                       & CONDUSR NSGN  ;
6934                    CALL  SWPDBL    & AM2901 MP,MP,RAMF,SUBS,AQ
6935   0057A    /                       & CARRYH        ;
6936                    CONT            & AM2901 ,,QREG,SUBS,ZQ
6937   0057B    /                       & CARRYH        ;
6938            ;
6939            ;       CHECK FOR MAGNITUDE OF SHIFT
6940            ;
6941            TADD2:  CONT            & AM2901 ,,QREG,SUBR,DQ
6942            /                       & CARRYL        ;
6943            /                       & IMM H#0010    ; SUBTRACT 16
6944   0057C    /                       & LUSRCOND      ;
```

```
LINE    ADDR    STATEMENT

6945                    CJP   TSHIFT     & AM2901 ,,NOP,ADD,AQ
6946    0057D   /                        & CONDUSR SGN     ; JP IF IN [0,15]
6947                    CONT               & AM2901 S6,S7,QREG,SUBR,DQ
6948            /                        & CARRYL          ;
6949            /                        & IMM H#0010      ; SUBTRACT 16
6950    0057E   /                        & LUSRCOND        ;
6951                    CJP   WRD1         & AM2901 S6,S7,RAMF,PASS,ZA
6952    0057F   /                        & CONDUSR SGN     ; JP IF IN [16,31]
6953                    CONT               & AM2901 S5,S7,QREG,SUBR,DQ
6954            /                        & CARRYL          ;
6955            /                        & IMM H#0010      ; SUBTRACT 16
6956    00580   /                        & LUSRCOND        ;
6957                    CJP   WRD2         & AM2901 S5,S7,RAMF,PASS,ZA
6958    00581   /                        & CONDUSR SGN     ; JP IF IN [32,47]
6959                    CONT               & AM2901 S4,S7,NOP,SUBR,DQ
6960            /                        & CARRYL          ;
6961            /                        & IMM H#0009      ; SUBTRACT 9
6962    00582   /                        & LUSRCOND        ;
6963                    CJP   DPACK        & AM2901 S4,S7,RAMF,PASS,ZA
6964    00583   /                        & CONDUSR NSGN    ; IF >56, SWAMP
6965                    CONT               & AM2901 S4,S4,RAMF,SUBR,AB
6966            /                        & CONDUSR SGN     ;
6967    00584   /                        & CARRYEXT        ; SIGN EXTEND IN S4
6968                    CONT               & AM2901 S4,S6,RAMF,PASS,ZA
6969    00585   /                        & SPNOP           ; SHIFT 3 WORDS
6970                    CONT               & AM2901 S4,S5,RAMF,PASS,ZA
6971    00586   /                        & SPNOP           ; AND SIGN EXTEND
6972                    JP    SHIFT1       & AM2901 ,,NOP,PASS,ZQ
6973    00587   /                        & LODUSR          ; USR = Q STATUS
6974            WRD2:   CALL  SIGNEXT      & AM2901 S4,S6,RAMF,PASS,ZA
6975    00588   /                        & LODUSR          ; SHIFT 2 WORDS
6976                    JP    TSHIFT       & AM2901 S4,S5,RAMF,PASS,ZA
6977    00589   /                        & SPNOP           ; AND SIGN EXTEND
6978            WRD1:   CONT               & AM2901 S5,S6,RAMF,PASS,ZA
6979    0058A   /                        & SPNOP           ; SHIFT ONE WORD
6980                    CALL  SIGNEXT      & AM2901 S4,S5,RAMF,PASS,ZA
6981    0058B   /                        & LODUSR          ; AND SIGN EXTEND
6982            TSHIFT: CONT               & AM2901 ,,QREG,ADD,DQ
6983            /                        & CARRYL          ;
6984            /                        & IMM H#0010      ; RESTORE PROPER SHIFT COUN
6985    0058C   /                        & LUSRCOND        ;
6986            SHIFT1: CJP   DBLADD       & AM2901 S7,S3,QREG,SUBR,ZQ
6987            /                        & CARRYH          ; DECREMENT Q
6988    0058D   /                        & CONDUSR Z       ; IF Q WAS 0, NO MORE SHIFT
6989            ;
6990            ;       If Q > 9, DO A LEFT SHIFT
6991            ;
6992                    CONT               & AM2901 ,,NOP,SUBR,DQ
6993            /                        & IMM H#0009      ;
6994            /                        & CARRYL          ; SUBTRACT 9 MORE
6995    0058E   /                        & LUSRCOND        ; (Q ALREADY DECREMENTED BY
6996                    CJP   LSHIFT       & AM2901 S7,S4,NOP,PASS,ZB
6997    0058F   /                        & CONDUSR NSGN    ; JP IF Q > 9;USR = S4
```

```
LINE    ADDR    STATEMENT

6999            ;       Residual is a right shift.
7000            ;       In two parts: first loop for S4 positive;  second
7001            ;       loop for S4 negative.  Counter loaded from Q lsb.
7002            ;
7003                    CJP   TADD11    & AM2901 S7,S7,RAMF,EXOR,AQ
7004    00590   /                       & CONDUSR SGN    ; JP IF NEG;PREPARE TO SWAP
7005                    CONT            & AM2901 S7,,QREG,EXOR,AQ
7006            /                       & LODMSR RESET   ; QREG = OLD S7
7007    00591   /                       & ENBLC          ; CLEAR MC
7008                    PUSH            & AM2901 S7,,NOP,EXOR,AQ
7009    00592   /                       & SPETC LDCTY    ; LDCT WITH OLD Q (MINUS 1)
7010                    CONT            & AM2901 ,S4,SRAMR,PASS,ZB
7011    00593   /                       & SHIFT B#1001   ;
7012                    CONT            & AM2901 ,S5,SRAMR,PASS,ZB
7013    00594   /                       & SHIFT B#1001   ;
7014                    RFCT            & AM2901 ,S6,SRAMQR,PASS,ZB
7015            /                       & SHIFT B#0100   ;
7016            /                       & LODMSR RESET   ;
7017    00595   /                       & ENBLC          ; CLEAR MC
7018                    JP    DBLADD    & AM2901 S7,S7,RAMF,PASS,ZQ
7019    00596   /                       & SPNOP          ; RESTORE S7
7020            ;
7021            ;       Right shift of negative operand.
7022            ;
7023            TADD11: CONT            & AM2901 S7,,QREG,EXOR,AQ
7024            /                       & LODMSR SET     ; Q = OLD S7
7025    00597   /                       & ENBLC          ; SET CARRY FOR SIGN EXTEND
7026                    PUSH            & AM2901 S7,S4,NOP,EXOR,AQ
7027    00598   /                       & SPETC LDCTY    ; LDCT WITH OLD Q
7028                    CONT            & AM2901 ,S4,SRAMR,PASS,ZB
7029    00599   /                       & SHIFT B#1001   ;
7030                    CONT            & AM2901 ,S5,SRAMR,PASS,ZB
7031    0059A   /                       & SHIFT B#1001   ;
7032                    RFCT            & AM2901 ,S6,SRAMQR,PASS,ZB
7033            /                       & SHIFT B#0100   ;
7034            /                       & LODMSR SET     ;
7035    0059B   /                       & ENBLC          ; SET MSR FOR SIGN EXTEND
7036                    JP    DBLADD    & AM2901 S7,S7,RAMF,PASS,ZQ
7037    0059C   /                       & SPNOP          ; RESTORE S7
```

```
LINE    ADDR    STATEMENT

7039            ;       Left shift: right shift one word, moving extra bits
7040            ;       into Q, and then left shift the amount we overdid it.
7041            ;
7042            LSHIFT: LDCT            & AM2901 ,S7,NOP,SUBS,DQ
7043            /                       & CARRYH        ; THIS INSTRUCTION LOADS CO
7044            /                       & IMMB H#0E     ;
7045    0059D   /                       & SPETC LDCTY   ; WITH 16-Q-1 FOR THE RFCT
7046                    CONT            & AM2901 S6,S7,QREG,PASS,ZB
7047    0059E   /                       & LODMSR RESET  ; Q = S7 ; MSR = CLEAR
7048                    CONT            & AM2901 S6,S7,RAMF,PASS,ZA
7049    0059F   /                       & SPNOP         ; S7 = S6
7050                    CONT            & AM2901 S5,S6,RAMF,PASS,ZA
7051    005A0   /                       & SPNOP         ; S6 = S5
7052                    CALL  SIGNEXT   & AM2901 S4,S5,RAMF,PASS,ZA
7053    005A1   /                       & LODUSR        ; S5 = S4; SIGN EXTEND S4
7054                    CPUSH           & AM2901 ,S7,NOP,ADD,AQ
7055    005A2   /                       & CONDMSR SGN   ; CPUSH NEVER LOADS COUNTER
7056                    CONT            & AM2901 ,S7,SRAMQL,PASS,ZB
7057    005A3   /                       & SHIFT B#1100  ;
7058                    CONT            & AM2901 ,S6,SRAML,PASS,ZB
7059    005A4   /                       & SHIFT B#1001  ;
7060                    CONT            & AM2901 ,S5,SRAML,PASS,ZB
7061    005A5   /                       & SHIFT B#1001  ;
7062                    RFCT            & AM2901 S7,S4,SRAML,PASS,ZB
7063    005A6   /                       & SHIFT B#1001  ;
```

```
LINE    ADDR    STATEMENT

7065            ;       Second operand now aligned.  Add the operands.
7066            ;
7067            DBLADD: CONT            & AM2901 S7,S3,RAMF,ADD,AB
7068            /                       & CARRYL        ;
7069    005A7   /                       & LODUSR        ;
7070                    CONT            & AM2901 S6,S2,RAMF,ADD,AB
7071    005A8   /                       & CARRYUC       ; PROPAGATE CARRY
7072                    CONT            & AM2901 S5,S1,RAMF,ADD,AB
7073    005A9   /                       & CARRYUC       ; PROPAGATE CARRY
7074                    CONT            & AM2901 S4,S0,RAMF,ADD,AB
7075    005AA   /                       & CARRYUC       ; PROPAGATE CARRY
7076                    CJP   DNORM     & AM2901 ,,NOP,ADD,AQ
7077    005AB   /                       & CONDUSR NOVR  ; CHECK FOR OVERFLOW
7078            ;
7079            ;       Overflow in addition.  Right shift and bump exponent.
7080            ;       Bit shifted into sign is complement of sign; it comes
7081            ;       from adding S0 to itself & loading carry inverted.
7082            ;
7083            DFIXOFL: CONT           & AM2901 S0,S0,NOP,ADD,AB
7084            /                       & CARRYL        ;
7085            /                       & LODMSR CI     ;
7086    005AC   /                       & ENBLC         ;
7087                    CONT            & AM2901 ,S0,SRAMR,PASS,ZB
7088    005AD   /                       & SHIFT B#1001  ;
7089                    CONT            & AM2901 ,S1,SRAMR,PASS,ZB
7090    005AE   /                       & SHIFT B#1001  ;
7091                    CONT            & AM2901 ,S2,SRAMR,PASS,ZB
7092    005AF   /                       & SHIFT B#1001  ;
7093                    CONT            & AM2901 ,S3,SRAMR,PASS,ZB
7094    005B0   /                       & SHIFT B#1001  ;
7095                    JP    DROUND    & AM2901 ,MP,RAMF,ADD,ZB
7096    005B1   /                       & CARRYH        ;
```

```
LINE    ADDR    STATEMENT

7098            ;       Normalize result.  First, by words.
7099            ;
7100            DNORM:  LDCT  DNCOUNT   & AM2901 S0,S4,RAMF,OR,ZA
7101            /                       & CARRYH        ;
7102    005B2   /                       & LODUSR        ;
7103            DNORM1: CJP   DNORM8    & AM2901 S0,S1,NOP,EXOR,AB
7104    005B3   /                       & CONDUSR UGT   ; IF NOT ZERO OR ONES
7105                    CJP   DNORM2    & AM2901 S0,S2,NOP,EXOR,AB
7106    005B4   /                       & CONDUSR Z     ;
7107            ;
7108            ;       Normalize one full word; decrement exponent by 16.
7109            ;
7110                    CONT            & AM2901 MP,MP,RAMF,SUBR,DA
7111            /                       & CARRYL        ;
7112    005B5   /                       & IMM H#0010    ;
7113                    CONT            & AM2901 S1,S0,RAMF,PASS,ZA
7114    005B6   /                       & SPNOP         ;
7115                    CONT            & AM2901 S2,S1,RAMF,PASS,ZA
7116    005B7   /                       & SPNOP         ;
7117                    JP    DNORM7    & AM2901 S3,S2,RAMF,PASS,ZA
7118    005B8   /                       & SPNOP         ;
7119            ;
7120            ;       Normalize two full words.  Decrement exponent by 32.
7121            ;
7122            DNORM2: CJP   DNORM3    & AM2901 S0,S3,NOP,EXOR,AB
7123    005B9   /                       & CONDUSR Z     ;
7124                    CONT            & AM2901 MP,MP,RAMF,SUBR,DA
7125            /                       & CARRYL        ;
7126    005BA   /                       & IMM H#0020    ;
7127                    CONT            & AM2901 S2,S0,RAMF,PASS,ZA
7128    005BB   /                       & SPNOP         ;
7129                    JP    DNORM6    & AM2901 S3,S1,RAMF,PASS,ZA
7130    005BC   /                       & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

7132            ;       Normalize three full words. Decrement exponent by 48.
7133            ;       If fourth word = signs, must be zero.
7134            ;
7135            DNORM3: CJP   DZERO     & AM2901 S3,S0,RAMF,PASS,ZA
7136    005BD   /                       & CONDUSR Z       ; IF 0, JP
7137                    CONT            & AM2901 MP,MP,RAMF,SUBR,DA
7138            /                       & CARRYL          ;
7139    005BE   /                       & IMM H#0030      ;
7140                    CONT            & AM2901 ,S1,RAMF,AND,ZB
7141    005BF   /                       & SPNOP           ;
7142            DNORM6: CONT            & AM2901 ,S2,RAMF,AND,ZB
7143    005C0   /                       & SPNOP           ;
7144            DNORM7: CONT            & AM2901 ,S3,RAMF,AND,ZB
7145    005C1   /                       & SPNOP           ;
7146            ;
7147            ;       Check that did not discard all of sign bits;
7148            ;       if no signs left, right shift one & fix sign.
7149            ;
7150                    CONT            & AM2901 S0,S4,NOP,EXOR,AB
7151    005C2   /                       & LODUSR          ;
7152                    CJP   DFIXOFL   & AM2901 ,,NOP,ADD,AQ
7153            /                       & CARRYL          ;
7154    005C3   /                       & CONDUSR SGN     ;
7155            ;
7156            ;       Bit normalize.  Determine if small, large or no shift.
7157            ;
7158            DNORM8: CONT            & AM2901 S0,S3,NOP,AND,DA
7159            /                       & IMM H#FF80      ;
7160    005C4   /                       & LUSRCOND        ;
7161                    CJP   DNORM12   & AM2901 ,S3,QREG,PASS,ZB
7162    005C5   /                       & CONDUSR Z       ;  Large, mant. positive.
7163                    CONT            & AM2901 S0,,NOP,OR,DA
7164            /                       & CARRYH          ;
7165            /                       & IMM H#007F      ;
7166    005C6   /                       & LUSRCOND        ; Carryout iff S0 .ne. -1
7167                    CJP   DNORM13   & AM2901 S0,S0,NOP,ADD,AB
7168            /                       & CARRYL          ;
7169    005C7   /                       & CONDUSR NC      ; Large, mant. negative.
```

```
LINE    ADDR    STATEMENT

7171            ;       Small or no shift.  Will shift one time too many.
7172            ;
7173                    CJP   DROUND     & AM2901 S3,S2,QREG,PASS,ZB
7174    005C8   /                        & CONDUSR OVR     ;
7175                    PUSH  DNCOUNT    & AM2901 S3,S3,RAMF,ADD,AB
7176           /                         & CARRYL          ;
7177           /                         & LODMSR          ; SHIFT S3 LEFT
7178    005C9  /                         & ENBLC           ; MC = BIT SHIFTED OUT
7179                    CONT             & AM2901 ,S1,SRAMQL,PASS,ZB
7180    005CA  /                         & SHIFT B#1100    ; SHIFT S6&S5 LEFT
7181                    CONT             & AM2901 S0,S0,RAMF,ADD,AB
7182           /                         & CARRYEXT        ; SHIFT LEFT AND ADD MC BIT
7183    005CB  /                         & CONDMSR C       ; LOAD USR WITH OVR STATUS
7184                    TWB              & AM2901 ,S3,SRAML,PASS,ZB
7185           /                         & CONDUSR OVR     ; TWB NEVER TAKES CTR=0 EXI
7186    005CC  /                         & SHIFT B#0000    ;
7187                    CONT             & AM2901 ,S3,SRAMR,PASS,ZB
7188    005CD  /                         & SHIFT B#1100    ; UNSHIFT S3
7189            ;
7190            ;       Undo the extra shift.  If more than one normalization
7191            ;       shift was required, the least 8 bits of the result
7192            ;       must be zero, so don't waste time trying to round it.
7193            ;       (Also adjust the exponent for the shift count.)
7194            ;
7195                    CONT             & AM2901 S0,S0,NOP,ADD,AB
7196           /                         & CARRYL          ;
7197           /                         & LODMSR CI       ;
7198    005CE  /                         & ENBLC           ;
7199                    CONT             & AM2901 ,S0,SRAMR,PASS,ZB
7200    005CF  /                         & SHIFT B#1001    ;
7201                    CONT             & AM2901 ,S1,SRAMQR,PASS,ZB
7202           /                         & SHIFT B#1100    ;
7203    005D0  /                         & LODMSR RESET    ;
7204                    JSRP  DNCOUNT    & AM2901 ,S2,RAMF,PASS,ZQ
7205    005D1  /                         & CONDMSR SGN     ; Condition always false
7206                    CONT             & AM2901 ,S3,SRAMR,PASS,ZB
7207    005D2  /                         & SHIFT B#1001    ;
7208                    LDCT  DPACK      & AM2901 S7,S7,NOP,SUBR,ZB
7209           /                         & CARRYH          ;
7210    005D3  /                         & LODUSR          ;
7211                    JRP   DROUND     & AM2901 S7,MP,RAMF,SUBR,AB
7212           /                         & CARRYL          ;
7213    005D4  /                         & CONDUSR Z       ;
```

```
LINE    ADDR    STATEMENT

7215            ;       Large shift.  Shift right until sign shifted into R5,
7216            ;       then left shift one word.  Since more than one bit of
7217            ;       normalization is required, the least 7 bits of R7
7218            ;       must be zero. Since
7219            ;       the right shift will be by 8 bits or less, we can get
7220            ;       by with a 4-word shift (and E); when the word shift
7221            ;       is done, E is put in the last word MSB; the rest of
7222            ;       the word is zero.  Two cases are handled seperately;
7223            ;       first, positive mantissa.
7224            ;
7225            ;       The shift is circular; since the least 7 bits of R7
7226            ;       are zero, correct sign bits will be shifted in on the
7227            ;       second thru eighth shifts.
7228            ;
7229            DNORM12: PUSH  DNCOUNT    & AM2901 ,S3,QREG,PASS,ZB
7230            /                         & LODMSR RESET    ;
7231    005D5   /                         & ENBLC           ;
7232                     CONT             & AM2901 ,S0,SRAMR,PASS,ZB
7233            /                         & SHIFT B#1001    ;
7234    005D6   /                         & LODMSR          ;
7235                     CONT             & AM2901 ,S1,SRAMR,PASS,ZB
7236    005D7   /                         & SHIFT B#1001    ;
7237                     TWB              & AM2901 ,S2,SRAMQR,PASS,ZB
7238            /                         & CONDMSR Z       ;
7239    005D8   /                         & SHIFT B#1100    ;
7240                     CONT             & AM2901 S1,S0,RAMF,PASS,ZA
7241    005D9   /                         & SPNOP           ;
7242                     JP     DNORM15    & AM2901 S2,S1,RAMF,PASS,ZA
7243    005DA   /                         & SPNOP           ;
```

```
LINE    ADDR    STATEMENT

7245            ;       Seperate loop for negative mansissa.
7246            ;       This loop begins in the middle & exits prematurely,
7247            ;       because the "ones" condition cannot be saved.  (The
7248            ;       counter was loaded at "DNORM")  The shift is logical,
7249            ;       with a "1" bit shifted in for a sign.  This is so
7250            ;       that when the rightmost sign is about to be shifted
7251            ;       into R5, the value in R4 will be -1, which is
7252            ;       detected by absence of "overflow" in an OR operation.
7253            ;
7254            DNORM13:  CALL  DNORM14     & AM2901 ,S3,QREG,PASS,ZB
7255            /                           & LODMSR SET      ;
7256    005DB   /                           & ENBLC           ;
7257                      CONT              & AM2901 ,S2,SRAMQR,PASS,ZB
7258            /                           & SHIFT B#0100    ;
7259            /                           & LODMSR SET      ;
7260    005DC   /                           & ENBLC           ;
7261            DNORM14:  CONT              & AM2901 ,S0,SRAMR,OR,ZB
7262            /                           & CARRYH          ;
7263            /                           & LODUSR          ;
7264    005DD   /                           & SHIFT B#1001    ;
7265                      TWB               & AM2901 ,S1,SRAMR,PASS,ZB
7266            /                           & SHIFT B#1001    ; TWB COUNTER NEVER = 0
7267    005DE   /                           & CONDUSR NOVR    ;
7268                      CONT              & AM2901 S1,S0,RAMF,PASS,ZA
7269    005DF   /                           & SPNOP           ;
7270                      CONT              & AM2901 S2,S1,SRAMQR,PASS,ZA
7271    005E0   /                           & SHIFT B#1100    ;
7272            DNORM15:  CONT              & AM2901 ,S3,SRAMR,AND,ZB
7273            /                           & SHIFT B#1001    ;
7274    005E1   /                           & LODMSR RESET    ;
7275                      JSRP              & AM2901 ,S2,RAMF,PASS,ZQ
7276    005E2   /                           & CONDMSR SGN     ; (Condition always false)
7277                      CONT              & AM2901 MP,MP,RAMF,SUBR,DA
7278            /                           & IMM H#0010      ;
7279    005E3   /                           & CARRYL          ;
7280                      JP    DPACK       & AM2901 S7,MP,RAMF,ADD,AB
7281    005E4   /                           & CARRYH          ;
```

```
LINE    ADDR    STATEMENT

7283            ;       Round mantissa: add 200B if +, 177B if -.
7284            ;
7285            DROUND: CONT            & AM2901 S0,S0,NOP,ADD,AB
7286            /                       & CARRYL        ;
7287    005E5   /                       & LODUSRCI      ;
7288                    CONT            & AM2901 S3,S3,RAMF,ADD,DA
7289            /                       & IMM H#007F    ;
7290    005E6   /                       & CARYUCOND     ; Causes CARRYUC
7291                    CONT            & AM2901 ,S2,RAMF,ADD,ZB
7292    005E7   /                       & CARRYUC       ;
7293                    CONT            & AM2901 ,S1,RAMF,ADD,ZB
7294    005E8   /                       & CARRYUC       ;
7295                    CONT            & AM2901 ,S0,RAMF,ADD,ZB
7296    005E9   /                       & CARRYUC       ;
7297            ;
7298            ;       Handle overflow & neg unnormalized after rounding.
7299            ;
7300                    CJP   DROUND1   & AM2901 S0,S0,NOP,ADD,AB
7301            /                       & CARRYL        ;
7302    005EA   /                       & CONDUSR NOVR  ;
7303                    CONT            & AM2901 ,S0,SRAMR,PASS,ZB
7304    005EB   /                       & SHIFT B#0000  ;
7305                    JP    DPACK     & AM2901 ,MP,RAMF,ADD,ZB
7306    005EC   /                       & CARRYH        ;
7307            DROUND1: CJP  DPACK     & AM2901 ,,NOP,ADD,AQ
7308    005ED   /                       & CONDUSR OVR   ;
7309                    CONT            & AM2901 ,S0,SRAML,PASS,ZB
7310    005EE   /                       & SHIFT B#0000  ;
7311                    CONT            & AM2901 ,MP,RAMF,SUBR,ZB
7312    005EF   /                       & CARRYH        ;
```

```
LINE    ADDR    STATEMENT

7314            ;       Check for overflow and underflow, and pack exponent.
7315            ;       See single precision microcode for algorithm.
7316            ;       At end, set E=0 which will be put in O.
7317            ;
7318            DPACK:  CONT            & AM2901 PC,S5,RAMF,ADD,DA
7319            /                       & CARRYL        ; S5 = NEXT INST
7320    005F0   /                       & IMM H#0002    ;
7321            DPACK1: CONT            & AM2901 MP,S4,RAMF,AND,DA
7322    005F1   /                       & IMM H#0080    ;
7323                    CONT            & AM2901 MP,S4,RAMF,ADD,AB
7324            /                       & CARRYL        ;
7325    005F2   /                       & LODUSR        ;
7326                    CONT            & AM2901 S4,S4,RAMF,ADD,AB
7327    005F3   /                       & CARRYUC       ;
7328                    CONT            & AM2901 S4,,NOP,AND,DA
7329            /                       & IMM H#FF00    ;
7330    005F4   /                       & LUSRCOND      ;
7331                    CJP   DOFLUFL   & AM2901 S3,MP,NOP,PASS,ZB
7332    005F5   /                       & CONDUSR NZ    ;
7333                    CONT            & AM2901 S3,S3,RAMF,AND,DA
7334    005F6   /                       & IMM H#FF00    ;
7335                    CONT            & AM2901 S4,S3,RAMF,OR,AB
7336            /                       & LODMSR RESET  ;
7337    005F7   /                       & ENBLC         ;
7338            ;
7339            ;       Store result.  (First, restore E in case interrupted.)
7340            ;
7341            DSTORE: CALL  INDRES    & AM2901 ,PC,NOP,SUBR,ZB
7342            /                       & CARRYH        ;
7343            /                       & LODMSR SWAPEO ;
7344            /                       & ENBLC & ENBLO ; SWAP E&O
7345            /                       & LDMAR         ;
7346    005F8   /                       & CREAD         ; READ DEF
7347            DSTORE1: CALL  INCQ     & AM2901 S0,TAB,NOP,PASS,ZA
7348            /                       & LDMDOR        ;
7349    005F9   /                       & DWRITE        ; WRITE FIRST WORD OF RESUL
7350                    CALL  INCQ      & AM2901 S1,TAB,NOP,PASS,ZA
7351            /                       & LDMDOR        ;
7352    005FA   /                       & DWRITE        ; WRITE SECOND WORD
7353                    CALL  INCQ      & AM2901 S2,TAB,NOP,PASS,ZA
7354            /                       & LDMDOR        ;
7355    005FB   /                       & DWRITE        ; WRITE THIRD WORD
7356                    CONT            & AM2901 S3,TAB,NOP,PASS,ZA
7357            /                       & LDMDOR        ;
7358    005FC   /                       & DWRITE        ; WRITE 4TH WORD
7359            ;
7360            ;       Exit; next instruction is at S5.
7361            ;
7362                    JZ              & AM2901 S5,PC,RAMA,ADD,ZA
7363            /                       & CARRYH        ;
7364            /                       & LDMAR         ;
7365    005FD   /                       & IFETCH        ;
```

```
LINE    ADDR    STATEMENT

7367            ;  Exponent correction table.  Before a normalize loop, the
7368            ;  counter is loaded with the address of "DCOUNT"; the counter
7369            ;  is decremented within the loop.  After the loop, a JSRP is
7370            ;  used to call the appropriate line of the table below, which
7371            ;  sets S7 to the number of loops executed, minus one.
7372            ;
7373                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7374    005FE   /                       & IMM H#0010      ;
7375                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7376    005FF   /                       & IMM H#000F      ;
7377                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7378    00600   /                       & IMM H#000E      ;
7379                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7380    00601   /                       & IMM H#000D      ;
7381                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7382    00602   /                       & IMM H#000C      ;
7383                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7384    00603   /                       & IMM H#000B      ;
7385                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7386    00604   /                       & IMM H#000A      ;
7387                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7388    00605   /                       & IMM H#0009      ;
7389                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7390    00606   /                       & IMM H#0008      ;
7391                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7392    00607   /                       & IMM H#0007      ;
7393                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7394    00608   /                       & IMM H#0006      ;
7395                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7396    00609   /                       & IMM H#0005      ;
7397                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7398    0060A   /                       & IMM H#0004      ;
7399                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7400    0060B   /                       & IMM H#0003      ;
7401                    RET             & AM2901 S7,S7,RAMF,PASS,DZ
7402    0060C   /                       & IMM H#0002      ;
7403            DNCOUNT2: RET           & AM2901 S7,S7,RAMF,PASS,DZ
7404    0060D   /                       & IMM H#0001      ;
7405            ;
7406            DNCOUNT1: EQU DNCOUNT2+1
7407            DNCOUNT:  EQU DNCOUNT2+2
```

```
LINE    ADDR    STATEMENT

7409            ;
7410            ;       Result = 0;  S1 = PC + 2  for exit address.
7411            ;
7412            DZERO:  CONT            & AM2901 PC,S5,RAMF,ADD,DA
7413            /                       & CARRYL          ;
7414    0060E   /                       & IMM H#0002      ;
7415                    CONT            & AM2901 ,S0,RAMF,AND,ZB
7416    0060F   /                       & SPNOP           ;
7417                    CONT            & AM2901 ,S1,RAMF,AND,ZB
7418    00610   /                       & SPNOP           ;
7419                    CONT            & AM2901 ,S2,RAMF,AND,ZB
7420    00611   /                       & SPNOP           ;
7421                    JP    DSTORE    & AM2901 ,S3,RAMF,AND,ZB
7422            /                       & LODMSR RESET    ;
7423    00612   /                       & ENBLC           ;
7424            ;
7425            ;       Handle double precision overflow and underflow.
7426            ;
7427            DOFLUFL: CJP  DOFL      & AM2901 S1,S1,RAMF,EXNOR,AB
7428    00613   /                       & CONDUSR NSGN    ;
7429                    CONT            & AM2901 S1,S1,RAMF,EXOR,AB
7430    00614   /                       & SPNOP           ;
7431            DOFL:   CONT            & AM2901 S1,S0,SRAMR,PASS,ZA
7432    00615   /                       & SHIFT B#0000    ;
7433                    CONT            & AM2901 S1,S3,SRAML,PASS,ZA
7434    00616   /                       & SHIFT B#0000    ;
7435                    JP    DSTORE    & AM2901 S1,S2,RAMF,PASS,ZA
7436            /                       & LODMSR SET      ;
7437            /                       & ENBLC           ;
7438    00617   /                       & LDMAR
```

```
LINE    ADDR    STATEMENT

7440            ;       Routine to set S4 to 0 or -1 depending on uN,
7441            ;       which must be set in the same line as the call.
7442            ;
7443            SIGNEXT: RET            & AM2901 S4,S4,RAMF,SUBR,AB
7444            /                       & CARRYEXT      ;
7445    00618   /                       & CONDUSR SGN   ;
7446            ;
7447    00619           FILLER
7448            ;
7449            ;
7450            ;       Routine to swap (S0,S1,S2,S3) with (S4,S5,S6,S7).
7451            ;       No other registers are affected.
7452            ;
7453            SWPDBL:  CONT           & AM2901 S0,S4,RAMF,EXOR,AB
7454    0061A   /                       & SPNOP         ;
7455                     CONT           & AM2901 S4,S0,RAMF,EXOR,AB
7456    0061B   /                       & SPNOP         ;
7457                     CONT           & AM2901 S0,S4,RAMF,EXOR,AB
7458    0061C   /                       & SPNOP         ;
7459                     CONT           & AM2901 S1,S5,RAMF,EXOR,AB
7460    0061D   /                       & SPNOP         ;
7461                     CONT           & AM2901 S5,S1,RAMF,EXOR,AB
7462    0061E   /                       & SPNOP         ;
7463                     CONT           & AM2901 S1,S5,RAMF,EXOR,AB
7464    0061F   /                       & SPNOP         ;
7465                     CONT           & AM2901 S2,S6,RAMF,EXOR,AB
7466    00620   /                       & SPNOP         ;
7467                     CONT           & AM2901 S6,S2,RAMF,EXOR,AB
7468    00621   /                       & SPNOP         ;
7469                     CONT           & AM2901 S2,S6,RAMF,EXOR,AB
7470    00622   /                       & SPNOP         ;
7471                     CONT           & AM2901 S3,S7,RAMF,EXOR,AB
7472    00623   /                       & SPNOP         ;
7473                     CONT           & AM2901 S7,S3,RAMF,EXOR,AB
7474    00624   /                       & SPNOP         ;
7475                     RET            & AM2901 S3,S7,RAMF,EXOR,AB
7476    00625   /                       & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

7478            ;*********************************************************************
7479            ;*                                                                   *
7480            ;*       Double precision multiply (.TMPY)                           *
7481            ;*                                                                   *
7482            ;*       JSB .TMPY                                                   *
7483            ;*       DEF PRODUCT                                                 *
7484            ;*       DEF MULTIPLICAND                                            *
7485            ;*       DEF MULTIPLIER                                              *
7486            ;*                                                                   *
7487            ;*********************************************************************
7488            ;
7489            ;       Get the operands.
7490            ;
7491            .TMPY:  CALL  DBLARG1   & AM2901 ,PC,NOP,PASS,ZB
7492            /                       & LDMAR         ;
7493   00626    /                       & CREAD         ; GET ARG1 IN (S0,S1,S2,S3)
7494                    CALL  DBLARG2   & AM2901 ,PC,NOP,ADD,ZB
7495            /                       & CARRYH        ;
7496            /                       & LDMAR         ;
7497   00627    /                       & CREAD         ; GET ARG2 IN (S4,S5,S6,S7)
7498            ;
7499            ;       Put result exponent and sign in MP; take abs values
7500            ;       CHECK FOR ARG1 = 0 OR ARG2 = 0
7501            ;
7502                    CONT            & AM2901 MP,MP,RAMF,ADD,AQ
7503   00628    /                       & CARRYH        ;
7504                    CONT            & AM2901 ,S0,NOP,PASS,ZB
7505   00629    /                       & LODMSR        ;
7506                    CCALL DNEGATE4  & AM2901 S0,S4,QREG,EXOR,AB
7507   0062A    /                       & CONDMSR SGN   ;
7508                    CJP   DZERO     & AM2901 ,S4,NOP,PASS,ZB
7509   0062B    /                       & CONDMSR Z     ;
7510                    CJP   DZERO     & AM2901 ,S4,NOP,PASS,ZB
7511   0062C    /                       & CONDUSR Z     ;
7512                    CCALL DNEGATE8  & AM2901 S7,S7,NOP,ADD,AQ
7513   0062D    /                       & CONDUSR SGN   ;
7514                    CONT            & AM2901 S7,MP,SRAMQL,PASS,ZB
7515   0062E    /                       & SHIFT B#0110  ;
7516            ;
7517            ;       Perform the unsigned multiply.  Register contents
7518            ;       are documented by noting which register(s) contain
7519            ;       which portions of the operands and result.
7520            ;       The 'S' register name prefix is dropped.
7521            ;
7522            ;       Initially:
7523            ;       op1 = (0,1,2,3)  op2 = (4,5,6,7)  res = (-,-,-,-)
7524            ;
7525            ;       The operands are referred to as x and y, and the
7526            ;       individual words are referred to as x1-x4 and y1-y4.
```

```
LINE    ADDR    STATEMENT

7528            ;       (0,1,2,3)    (4,5,6,7)    (-,-,-,-)
7529            ;
7530            ;       x1 * y4
7531            ;
7532                    CONT            & AM2901 S7,S7,RAMA,PASS,DZ
7533    0062F   /                       & SPRD RL4       ;
7534                    CONT            & AM2901 S7,S7,RAMA,PASS,DZ
7535    00630   /                       & SPRD RL4       ;
7536                    LDCT  H#007     & AM2901 ,S7,QREG,PASS,ZB
7537    00631   /                       & LODMSR RESET   ; MSR = 0
7538                    CPUSH           & AM2901 ,S7,SRAMQR,AND,ZB
7539            /                       & CONDMSR SGN    ; Always fails.
7540    00632   /                       & SHIFT B#0110   ; S7 = 0; SET Q0BUF
7541                    RFCT            & AM2901 MPY8,S7,SRAMQR,ADD,AB
7542            /                       & CARRYL         ;
7543    00633   /                       & SHIFT B#1011   ;
7544            ;
7545            ;       (0,1,2,3)    (4,5,6,-)    (-,-,-,7)
7546            ;
7547                    CONT            & AM2901 S0,S4,RAMF,EXOR,AB
7548    00634   /                       & SPNOP          ;
7549                    CONT            & AM2901 S4,S0,RAMF,EXOR,AB
7550    00635   /                       & SPNOP          ;
7551                    CONT            & AM2901 S0,S4,RAMF,EXOR,AB
7552    00636   /                       & SPNOP          ;
7553            ;
7554            ;       (4,1,2,3)    (0,5,6,-)    (-,-,-,7)
7555            ;
7556            ;       x4 * y1
7557            ;
7558                    CONT            & AM2901 S3,S3,RAMA,PASS,DZ
7559    00637   /                       & SPRD RL4       ;
7560                    CONT            & AM2901 S3,S3,RAMA,PASS,DZ
7561    00638   /                       & SPRD RL4       ;
7562                    LDCT  H#007     & AM2901 ,S3,QREG,PASS,ZB
7563    00639   /                       & LODMSR RESET   ; MSR = 0
7564                    CPUSH           & AM2901 ,S3,SRAMQR,AND,ZB
7565            /                       & CONDMSR SGN    ; Always fails.
7566    0063A   /                       & SHIFT B#0110   ; S3 = 0; SET Q0BUF
7567                    RFCT            & AM2901 MPY8,S3,SRAMQR,ADD,AB
7568            /                       & CARRYL         ;
7569    0063B   /                       & SHIFT B#1011   ;
```

```
LINE    ADDR    STATEMENT

7571            ;       (4,1,2,-)    (0,5,6,-)    (-,-,-,3+7)
7572            ;
7573                    CONT             & AM2901 S7,S3,RAMF,ADD,AB
7574            /                        & CARRYL          ;
7575            /                        & LODMSR          ;
7576    0063C   /                        & ENBLC           ;
7577            ;
7578            ;       (4,1,2,-)    (0,5,6,-)    (-,-,Mc,3)
7579            ;
7580            ;       x3 * y1 + S3
7581            ;
7582                    LDCT  H#00F      & AM2901 S3,S2,QREG,PASS,ZB
7583    0063D   /                        & LODMSR RESET    ; MSR = 0
7584                    CPUSH            & AM2901 S3,S3,SRAMQR,ADD,AB
7585            /                        & CARRYL          ;
7586            /                        & CONDMSR SGN     ; Always fails.
7587    0063E   /                        & SHIFT B#1110    ; SET Q0BUF, SAVE S3
7588                    RFCT             & AM2901 MPY8,S3,SRAMQR,ADD,AB
7589            /                        & CARRYL          ;
7590    0063F   /                        & SHIFT B#1011    ;
7591            ;
7592            ;       (4,1,2,-)    (0,5,6,-)    (-,-,3+Mc,Q)
7593            ;
7594                    CONT             & AM2901 ,S7,RAMF,PASS,ZQ
7595    00640   /                        & SPNOP           ;
7596                    CONT             & AM2901 S0,S5,RAMF,EXOR,AB
7597    00641   /                        & SPNOP           ;
7598                    CONT             & AM2901 S5,S0,RAMF,EXOR,AB
7599    00642   /                        & SPNOP           ;
7600                    CONT             & AM2901 S0,S5,RAMF,EXOR,AB
7601    00643   /                        & SPNOP           ;
7602            ;
7603            ;       (4,1,2,-)    (5,0,6,-)    (-,-,3+Mc,7)
7604            ;
7605            ;       x3 * y2
7606            ;
7607                    LDCT  H#00F      & AM2901 ,S2,QREG,PASS,ZB
7608    00644   /                        & LODMSR RESET    ; MSR = 0
7609                    CPUSH            & AM2901 ,S2,SRAMQR,AND,ZB
7610            /                        & CONDMSR SGN     ; Always fails.
7611    00645   /                        & SHIFT B#0110    ; S2 = 0; SET Q0BUF
7612                    RFCT             & AM2901 MPY8,S2,SRAMQR,ADD,AB
7613            /                        & CARRYL          ;
7614    00646   /                        & SHIFT B#1011    ;
```

```
LINE    ADDR    STATEMENT

7616            ;       (4,1,-,-)    (5,0,6,-)    (-,-,3+Mc,7+2)
7617            ;
7618                    CONT            & AM2901 S2,S7,RAMF,ADD,AB
7619            /                       & CARRYL         ;
7620    00647   /                       & LODUSR         ;
7621                    CONT            & AM2901 ,S3,RAMF,ADD,ZB
7622    00648   /                       & CARRYUC        ; NEVER GENERATES CARRY ?
7623                    CONT            & AM2901 S0,S1,RAMF,EXOR,AB
7624    00649   /                       & SPNOP          ;
7625                    CONT            & AM2901 S1,S0,RAMF,EXOR,AB
7626    0064A   /                       & SPNOP          ;
7627                    CONT            & AM2901 S0,S1,RAMF,EXOR,AB
7628    0064B   /                       & SPNOP          ;
7629            ;
7630            ;       (4,0,-,-)    (5,1,6,-)    (-,-,3+Mc,7)
7631            ;
7632            ;       x2 * y3
7633            ;
7634                    LDCT  H#00F     & AM2901 ,S6,QREG,PASS,ZB
7635    0064C   /                       & LODMSR RESET   ; MSR = 0
7636                    CPUSH           & AM2901 ,S2,SRAMQR,AND,ZB
7637            /                       & CONDMSR SGN    ; Always fails.
7638    0064D   /                       & SHIFT B#0110   ; S2 = 0; SET Q0BUF
7639                    RFCT            & AM2901 MPY8,S2,SRAMQR,ADD,AB
7640            /                       & CARRYL         ;
7641    0064E   /                       & SHIFT B#1011   ;
7642            ;
7643            ;       (4,0,-,-)    (5,1,6,-)    (-,-,3+Mc,7+2)
7644            ;
7645                    CONT            & AM2901 S0,S4,RAMF,EXOR,AB
7646    0064F   /                       & SPNOP          ;
7647                    CONT            & AM2901 S4,S0,RAMF,EXOR,AB
7648    00650   /                       & SPNOP          ;
7649                    CONT            & AM2901 S0,S4,RAMF,EXOR,AB
7650    00651   /                       & SPNOP          ;
```

```
LINE    ADDR    STATEMENT

7652            ;       (0,4,-,-)    (5,1,6,-)    (-,-,3+Mc,7+2)
7653            ;
7654            ;       x1 * y3 + S2
7655            ;
7656                    LDCT  H#00F      & AM2901 ,S6,QREG,PASS,ZB
7657    00652   /                        & LODMSR RESET   ; MSR = 0
7658                    CPUSH            & AM2901 S2,S2,SRAMQR,ADD,AB
7659            /                        & CARRYL         ;
7660            /                        & CONDMSR SGN    ; Always fails.
7661    00653   /                        & SHIFT B#1110   ; ADD S2; SET Q0BUF
7662                    RFCT             & AM2901 MPY8,S2,SRAMQR,ADD,AB
7663            /                        & CARRYL         ;
7664    00654   /                        & SHIFT B#1011   ;
7665            ;
7666            ;       (0,4,-,-)    (5,1,-,-)    (-,-,3+2+Mc,7+Q)
7667            ;
7668                    CONT             & AM2901 S7,S7,RAMF,ADD,AQ
7669            /                        & CARRYL         ;
7670    00655   /                        & LODUSR         ;
7671                    CONT             & AM2901 S3,S2,RAMF,ADD,ZB
7672    00656   /                        & CARRYUC        ; NEVER GENERATES CARRY
7673                    CONT             & AM2901 S3,S2,RAMF,ADD,AB
7674            /                        & CARRYEXT       ;
7675    00657   /                        & CONDMSR C      ;
7676            ;
7677            ;       (0,4,-,-)    (5,1,-,-)    (-,Uc,2,7)
7678            ;
7679            ;       x1 * y2 + S2
7680            ;
7681                    LDCT  H#00F      & AM2901 ,S1,QREG,PASS,ZB
7682            /                        & LODMSR SWAPUM  ; SET MC FROM PREV ADD
7683    00658   /                        & ENBLC          ; QREG = S1
7684                    CPUSH            & AM2901 S2,S2,SRAMQR,ADD,AB
7685            /                        & CONDUSR SGN    ; Always fails.
7686            /                        & CARRYL         ; (USR = OLD MSR = RESET)
7687    00659   /                        & SHIFT B#1110   ; ADD S2; SET Q0BUF
7688                    RFCT             & AM2901 MPY8,S2,SRAMQR,ADD,AB
7689            /                        & CARRYL         ;
7690    0065A   /                        & SHIFT B#1011   ;
7691            ;
7692            ;       (0,4,-,-)    (5,1,-,-)    (-,2+Mc,Q,7)
7693            ;
7694                    CONT             & AM2901 ,S3,RAMF,PASS,ZQ
7695    0065B   /                        & SPNOP          ;
```

```
LINE    ADDR    STATEMENT

7697            ;       (0,4,-,-)    (5,1,-,-)    (-,2+Mc,3,7)
7698            ;
7699            ;       x1 * y1 + S2
7700            ;
7701                    LDCT  H#00F      & AM2901 ,S5,QREG,PASS,ZB
7702    0065C   /                        & LODMSR RESET    ; CLEAR MSR
7703                    CPUSH            & AM2901 S2,S2,SRAMQR,ADD,AB
7704            /                        & CARRYL          ;
7705            /                        & CONDMSR SGN     ; Always fails.
7706    0065D   /                        & SHIFT B#1110    ; ADD S2; SET Q0BUF
7707                    RFCT             & AM2901 MPY8,S2,SRAMQR,ADD,AB
7708            /                        & CARRYL          ;
7709    0065E   /                        & SHIFT B#1011    ;
7710            ;
7711            ;       (-,4,-,-)    (5,1,-,-)    (2,Q+Mc,3,7)
7712            ;
7713                    CONT             & AM2901 ,S6,RAMF,ADD,ZQ
7714            /                        & CARRYEXT        ; ADD IN PREV MC
7715    0065F   /                        & CONDMSR C       ;
7716                    CONT             & AM2901 S4,S0,RAMF,PASS,ZA
7717            /                        & LODMSR SWAPUM   ;
7718    00660   /                        & ENBLC           ; LOAD MC FROM PREV LINE
7719            ;
7720            ;       (-,0,-,-)    (5,1,-,-)    (2+Mc,6,3,7)
7721            ;
7722            ;       x2 * y2 + S7
7723            ;
7724                    LDCT  H#00F      & AM2901 S7,S1,QREG,PASS,ZB
7725    00661   /                        & LODMSR RESET    ; MSR = 0
7726                    CPUSH            & AM2901 S7,S7,SRAMQR,ADD,AB
7727            /                        & CARRYL          ;
7728            /                        & CONDMSR SGN     ; Always fails.
7729    00662   /                        & SHIFT B#1110    ; ADD S7; SET Q0BUF
7730                    RFCT             & AM2901 MPY8,S7,SRAMQR,ADD,AB
7731            /                        & CARRYL          ;
7732    00663   /                        & SHIFT B#1011    ;
7733            ;
7734            ;       (-,0,-,-)    (5,-,-,-)    (2+Mc, 6, 3+7, Q)
7735            ;
7736                    CONT             & AM2901 ,S4,RAMF,PASS,ZQ
7737    00664   /                        & SPNOP           ;
```

```
LINE    ADDR    STATEMENT

7739            ;       (-,0,-,-)    (5,-,-,-)    (2+Mc, 6, 3+7, 4)
7740            ;
7741            ;       x2 * y1 + S7
7742            ;
7743                    LDCT  H#00F      & AM2901 S7,S5,QREG,PASS,ZB
7744    00665   /                        & LODMSR RESET   ; MSR = 0
7745                    CPUSH            & AM2901 S7,S7,SRAMQR,ADD,AB
7746            /                        & CARRYL         ;
7747            /                        & CONDMSR SGN    ; Always fails.
7748    00666   /                        & SHIFT B#1110   ; ADD S7; SET Q0BUF
7749                    RFCT             & AM2901 MPY8,S7,SRAMQR,ADD,AB
7750            /                        & CARRYL         ;
7751    00667   /                        & SHIFT B#1011   ;
7752            ;
7753            ;       (-,-,-,-)    (-,-,-,-)    (2+Mc, 6+7, 3+Q, 4)
7754            ;
7755                    CONT             & AM2901 S2,S0,RAMF,ADD,ZA
7756            /                        & CARRYEXT       ;
7757    00668   /                        & CONDMSR C      ;
7758                    CONT             & AM2901 S3,S2,RAMF,ADD,AQ
7759            /                        & CARRYL         ;
7760    00669   /                        & LODUSR         ;
7761                    CONT             & AM2901 S7,S6,RAMF,ADD,AB
7762    0066A   /                        & CARRYUC        ;
7763                    CONT             & AM2901 S6,S0,RAMF,ADD,ZB
7764    0066B   /                        & CARRYUC        ;
7765                    CONT             & AM2901 S6,S1,RAMF,PASS,ZA
7766    0066C   /                        & SPNOP          ;
7767                    CONT             & AM2901 S4,S3,RAMF,PASS,ZA
7768    0066D   /                        & SPNOP          ;
```

```
LINE    ADDR    STATEMENT

7770            ;         (-,-,-,-)    (-,-,-,-)    (0, 1, 2, 3)
7771            ;
7772            ;         If operand signs differed, negate result.
7773            ;         Also remove saved sign bit from MP.
7774            ;
7775            TMPYEXIT: CONT             & AM2901 ,MP,SRAMQR,ADD,ZB
7776            /                          & CARRYL         ;
7777   0066E    /                          & SHIFT B#1110   ;
7778                      LDCT  DROUND     & AM2901 ,,NOP,PASS,ZQ
7779   0066F    /                          & LODUSR         ;
7780                      CCALL DNEGATE4   & AM2901 ,MP,RAMF,ADD,ZB
7781            /                          & CONDUSR SGN    ;
7782   00670    /                          & CARRYH         ;
7783            ;
7784            ;         Normalize.  Zero to two shifts required.
7785            ;
7786                      JP    TMPYEX2    & AM2901 S0,S0,NOP,ADD,AB
7787            /                          & CARRYL         ;
7788   00671    /                          & LODUSR         ;
7789            TMPYEX1:  CONT             & AM2901 ,S3,SRAML,PASS,ZB
7790   00672    /                          & SHIFT B#0000   ;
7791                      CONT             & AM2901 ,S2,SRAML,PASS,ZB
7792   00673    /                          & SHIFT B#1001   ;
7793                      CONT             & AM2901 ,S1,SRAML,PASS,ZB
7794   00674    /                          & SHIFT B#1001   ;
7795                      CONT             & AM2901 ,S0,SRAML,PASS,ZB
7796   00675    /                          & SHIFT B#1001   ;
7797                      CONT             & AM2901 S0,S0,NOP,ADD,AB
7798            /                          & CARRYL         ;
7799   00676    /                          & LODUSR         ;
7800            TMPYEX2:  JRP   TMPYEX1    & AM2901 ,MP,RAMF,SUBR,ZB
7801            /                          & CARRYH         ;
7802   00677    /                          & CONDUSR NOVR   ; Fail => DROUND
```

```
LINE    ADDR    STATEMENT

7804            ;       Routine to negate (S0,S1,S2,S3);
7805            ;       no overflow correction or normalization done.
7806            ;
7807            DNEGATE4: CONT          & AM2901 ,S3,RAMF,SUBS,ZB
7808            /                       & CARRYH        ;
7809    00678   /                       & LODUSR        ;
7810                      CONT          & AM2901 ,S2,RAMF,SUBS,ZB
7811    00679   /                       & CARRYUC       ;
7812                      CONT          & AM2901 ,S1,RAMF,SUBS,ZB
7813    0067A   /                       & CARRYUC       ;
7814                      RET           & AM2901 ,S0,RAMF,SUBS,ZB
7815    0067B   /                       & CARRYUC       ;
7816            ;
7817            ;       Routine to negate (S4,S5,S6,S7);
7818            ;       no overflow correction or normalization done.
7819            ;
7820            DNEGATE8: CONT          & AM2901 ,S7,RAMF,SUBS,ZB
7821            /                       & CARRYH        ;
7822    0067C   /                       & LODUSR        ;
7823                      CONT          & AM2901 ,S6,RAMF,SUBS,ZB
7824    0067D   /                       & CARRYUC       ;
7825                      CONT          & AM2901 ,S5,RAMF,SUBS,ZB
7826    0067E   /                       & CARRYUC       ;
7827                      RET           & AM2901 S7,S4,RAMF,SUBS,ZB
7828    0067F   /                       & CARRYUC       ;
```

```
LINE    ADDR    STATEMENT

7830            ;*****************************************************************
7831            ;*                                                               *
7832            ;*       Double precision divide (.TDIV)                         *
7833            ;*                                                               *
7834            ;*       JSB .TDIV                                               *
7835            ;*       DEF RESULT                                              *
7836            ;*       DEF ARG1                                                *
7837            ;*       DEF ARG2                                                *
7838            ;*                                                               *
7839            ;*****************************************************************
7840            ;
7841            ;       Get the operands.
7842            ;
7843            .TDIV:  CALL  DBLARG1   & AM2901 ,PC,NOP,PASS,ZB
7844            /                       & LDMAR         ;
7845   00680    /                       & CREAD         ; GET ARG1 INTO (S0,S1,S2,S
7846                    CALL  DBLARG2   & AM2901 ,PC,NOP,ADD,ZB
7847            /                       & CARRYH        ;
7848            /                       & LDMAR         ;
7849   00681    /                       & CREAD         ; GET ARG2 INTO (S4,S5,S6,S
7850            ;
7851            ;       Put result exponent & sign in MP; take abs values.
7852            ;
7853                    CONT            & AM2901 MP,MP,RAMF,SUBS,AQ
7854   00682    /                       & CARRYH        ;
7855                    CONT            & AM2901 ,S4,NOP,PASS,ZB
7856   00683    /                       & LODMSR        ;
7857                    CCALL DNEGATE8  & AM2901 S0,S4,QREG,EXOR,AB
7858   00684    /                       & CONDMSR SGN   ;
7859                    CJP   TDIV9     & AM2901 ,S0,NOP,PASS,ZB
7860   00685    /                       & CONDMSR Z     ;
7861                    CCALL DNEGATE4  & AM2901 ,MP,RAMF,ADD,ZB
7862            /                       & CARRYH        ;
7863   00686    /                       & CONDUSR SGN   ;
7864                    CONT            & AM2901 ,MP,SRAMQL,PASS,ZB
7865   00687    /                       & SHIFT B#0110  ;
```

```
LINE    ADDR    STATEMENT

7867            ;       Right shift both by one bit: avoid certain end cases.
7868            ;
7869                    CONT            & AM2901 ,S0,SRAMR,PASS,ZB
7870    00688   /                       & SHIFT B#0010   ;
7871                    CONT            & AM2901 ,S1,SRAMR,PASS,ZB
7872    00689   /                       & SHIFT B#1001   ;
7873                    CONT            & AM2901 ,S2,SRAMR,PASS,ZB
7874    0068A   /                       & SHIFT B#1001   ;
7875                    CONT            & AM2901 ,S3,SRAMR,PASS,ZB
7876    0068B   /                       & SHIFT B#1001   ;
7877                    CONT            & AM2901 ,S4,SRAMR,PASS,ZB
7878    0068C   /                       & SHIFT B#0010   ;
7879                    CONT            & AM2901 ,S5,SRAMR,PASS,ZB
7880    0068D   /                       & SHIFT B#1001   ;
7881                    CONT            & AM2901 ,S6,SRAMR,PASS,ZB
7882    0068E   /                       & SHIFT B#1001   ;
7883                    CONT            & AM2901 ,S7,SRAMR,PASS,ZB
7884    0068F   /                       & SHIFT B#1001   ;
```

```
LINE    ADDR    STATEMENT

7886            ;       First divide loop.  Start with Mn=0, Mc=0, CT=1.
7887            ;       Exchange R4 and R7 so that first add/subtract of
7888            ;       each loop can use DIV, so that the CJP can overlap
7889            ;       the first operation of each loop.  The two parts of
7890            ;       the loop must be seperate because the SUBR operation
7891            ;       complements carry-in, so the carry propagate for
7892            ;       ADD and for SUBR are coded differently.
7893            ;
7894                    CONT            & AM2901 S3,S0,QREG,PASS,ZB
7895   00690    /                       & LODUSR        ;
7896                    CJP   DZERO     & AM2901 S3,S0,RAMF,PASS,ZA
7897   00691    /                       & CONDUSR Z     ;
7898                    CONT            & AM2901 ,S3,RAMF,PASS,ZQ
7899            /                       & LODMSR RESET  ;
7900   00692    /                       & ENBLC         ;
7901                    PUSH  H#00F     & AM2901 S7,,QREG,AND,ZQ
7902   00693    /                       & CONDMSR C     ; Set CCBUF
7903                    CJP   TDIV1     & AM2901 S7,DIV,SRAMQL,SUBR,AB
7904            /                       & CARRYL        ;
7905            /                       & CONDMSR C     ;
7906   00694    /                       & SHIFT B#1100  ;
7907                    CONT            & AM2901 S6,S2,SRAML,SUBR,AB
7908            /                       & CARRYNUC      ;
7909   00695    /                       & SHIFT B#1001  ;
7910                    CONT            & AM2901 S5,S1,SRAML,SUBR,AB
7911            /                       & CARRYUC       ;
7912   00696    /                       & SHIFT B#1001  ;
7913                    CONT            & AM2901 S0,S0,NOP,SUBR,AB
7914   00697    /                       & CARRYUC       ; INVERT UC.
7915                    RFCT            & AM2901 S4,S3,SRAML,SUBR,AB
7916            /                       & UCDIVCND      ;
7917   00698    /                       & SHIFT B#1001  ;
7918                    JP    TDIV2     & AM2901 S2,S7,RAMF,EXNOR,ZQ
7919   00699    /                       & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

7921            ;       Force LSB of $ to be 1100, matching shift code.
7922            ;
7923            TDIVSKIP: SET   H#FFC-$
7924                      DUP   TDIVSKIP_AND_H#00F
7925    0069A             FILLER
7925    0069B             FILLER
7926            ;
7927            TDIV1:    CONT            & AM2901 S6,S2,SRAML,ADD,AB
7928            /                         & CARRYUC       ;
7929    0069C   /                         & SHIFT B#1001  ;
7930                      CONT            & AM2901 S5,S1,SRAML,ADD,AB
7931            /                         & CARRYUC       ;
7932    0069D   /                         & SHIFT B#1001  ;
7933                      RFCT            & AM2901 S4,S3,SRAML,ADD,AB
7934            /                         & UCDIVCND      ;
7935    0069E   /                         & SHIFT B#1001  ;
7936                      CONT            & AM2901 S2,S7,RAMF,EXNOR,ZQ
7937    0069F   /                         & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

7939            ;       Second divide loop.
7940            ;
7941            TDIV2:  CONT            & AM2901 S2,S0,RAMF,PASS,ZA
7942    006A0   /                       & SPNOP         ;
7943                    PUSH  H#00F     & AM2901 S6,,QREG,AND,ZQ
7944    006A1   /                       & CONDMSR C     ; Set CC.
7945                    CJP   TDIV3     & AM2901 S6,DIV,SRAMQL,SUBR,AB
7946            /                       & CARRYL        ;
7947            /                       & CONDMSR C     ;
7948    006A2   /                       & SHIFT B#1100  ;
7949                    CONT            & AM2901 S5,S1,SRAML,SUBR,AB
7950            /                       & CARRYNUC      ;
7951    006A3   /                       & SHIFT B#1001  ;
7952                    RFCT            & AM2901 S4,S3,SRAML,SUBR,AB
7953            /                       & UCDIVCND      ;
7954    006A4   /                       & SHIFT B#1001  ;
7955                    JP    TDIV4     & AM2901 S4,S6,RAMF,EXNOR,ZQ
7956    006A5   /                       & SPNOP         ,
7957            ;
7958            ;       Force LSB of $ to be 1100, matching shift code.
7959            ;
7960            TDIVSKIP: SET   H#FFC-$
7961                    DUP   TDIVSKIP_AND_H#00F
7962    006A6           FILLER
7962    006A7           FILLER
7962    006A8           FILLER
7962    006A9           FILLER
7962    006AA           FILLER
7962    006AB           FILLER
7963            ;
7964            TDIV3:  CONT            & AM2901 S5,S1,SRAML,ADD,AB
7965            /                       & CARRYUC       ;
7966    006AC   /                       & SHIFT B#1001  ;
7967                    RFCT            & AM2901 S4,S3,SRAML,ADD,AB
7968            /                       & UCDIVCND      ;
7969    006AD   /                       & SHIFT B#1001  ;
7970                    JP    TDIV4     & AM2901 S4,S6,RAMF,EXNOR,ZQ
7971    006AE   /                       & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

7973            ;       Third divide loop.  Note that S2 is set to .not.S4,
7974            ;       which is used in subtract part to make the carry
7975            ;       propagate end up on a carryuc, as it must.
7976            ;
7977            TDIV4:  CONT            & AM2901 S4,S2,RAMF,EXNOR,ZA
7978    006AF   /                       & SPNOP         ;
7979                    CONT            & AM2901 S1,S0,RAMF,PASS,ZA
7980    006B0   /                       & SPNOP         ;
7981                    PUSH  H#00F     & AM2901 S5,,QREG,AND,ZQ
7982    006B1   /                       & CONDMSR C     ; Set CC.
7983                    CJP   TDIV5     & AM2901 S5,DIV,SRAMQL,SUBR,AB
7984            /                       & CARRYL        ;
7985            /                       & CONDMSR C     ;
7986    006B2   /                       & SHIFT B#1100  ;
7987                    RFCT            & AM2901 S2,S3,SRAML,ADD,AB
7988            /                       & UCDIVCND      ;
7989    006B3   /                       & SHIFT B#1001  ;
7990                    JP    TDIV6     & AM2901 S3,S2,RAMF,EXNOR,ZQ
7991    006B4   /                       & SPNOP         ;
7992            ;
7993            ;       Force LSB of $ to be 1100, matching shift code.
7994            ;
7995            TDIVSKIP: SET   H#FFC-$
7996                    DUP   TDIVSKIP_AND_H#00F
7997    006B5           FILLER
7997    006B6           FILLER
7997    006B7           FILLER
7997    006B8           FILLER
7997    006B9           FILLER
7997    006BA           FILLER
7997    006BB           FILLER
7998            ;
7999            TDIV5:  RFCT            & AM2901 S4,S3,SRAML,ADD,AB
8000            /                       & UCDIVCND      ;
8001    006BC   /                       & SHIFT B#1001  ;
8002                    JP    TDIV6     & AM2901 S3,S2,RAMF,EXNOR,ZQ
8003    006BD   /                       & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

8005            ;       Fourth divide loop.
8006            ;
8007            TDIV6:  CONT            & AM2901 S3,S0,RAMF,PASS,ZA
8008    006BE   /                       & SPNOP         ;
8009                    PUSH  H#00F     & AM2901 S4,,QREG,AND,ZQ
8010    006BF   /                       & CONDMSR C     ; Set up CC.
8011                    RFCT            & AM2901 S4,DIV,SRAMQL,SUBR,AB
8012            /                       & CARRYL        ;
8013            /                       & DIVCOND       ;
8014    006C0   /                       & SHIFT B#1100  ;
8015            ;
8016            ;       Put result in (S0,S1,S2,S3) & finish up.
8017            ;
8018                    CONT            & AM2901 S7,S0,RAMF,EXOR,DA
8019    006C1   /                       & IMM H#8000    ; MSB was wrong.
8020                    CONT            & AM2901 S6,S1,RAMF,PASS,ZA
8021    006C2   /                       & SPNOP         ;
8022                    JP    TMPYEXIT  & AM2901 ,S3,RAMF,EXNOR,ZQ
8023    006C3   /                       & SPNOP         ;
8024            ;
8025            ;
8026            ;       Divide by zero.  Take overflow exit.
8027            ;
8028            TDIV9:  CONT            & AM2901 ,MP,RAMF,EXNOR,DZ
8029    006C4   /                       & IMM H#F000    ;
8030                    JP    DPACK     & AM2901 ,,NOP,ADD,AQ
8031    006C5   /                       & SPNOP         ;
```

```
LINE    ADDR    STATEMENT

8033            ;*********************************************************************
8034            ;*                                                                   *
8035            ;*        VMA Microcode Routines                                     *
8036            ;*                                                                   *
8037            ;* - VMA pointer:                                                    *
8038            ;*                                                                   *
8039            ;*                                                                   *
8040            ;|       A  Register              |       B  Register               |
8041            ;+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
8042            ;|L|0 0 0 0 0| VMA Seg # |   PTE  Index      | Log Page Offset   |
8043            ;+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
8044            ;*                                                                   *
8045            ;* - L is local reference bit.  If L is set, then B register is      *
8046            ;*   normal address.  Simply resolve B for indirects and return      *
8047            ;*   address in B.                                                   *
8048            ;*                                                                   *
8049            ;* - The PTE index is an index into a table of VMA pages.  The       *
8050            ;*   page number of this table is contained in location 5.           *
8051            ;*   Access to the PTE table is by reading loc 5, loading            *
8052            ;*   contents into reg 31 of MAPX, then using 011111B as the         *
8053            ;*   upper bits of an address and the PTE Index as the lower         *
8054            ;*   bits.                                                           *
8055            ;*                                                                   *
8056            ;* - The PTE table entries can have three forms:                     *
8057            ;*                                                                   *
8058            ;*             15        10 9                  0   Circumstance    *
8059            ;*             +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                     *
8060            ;*    Case A   | VMA suit  |   Page offset     |  Normal             *
8061            ;*             +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                     *
8062            ;*                                                                   *
8063            ;*             +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                     *
8064            ;*    Case B   | VMA suit  |0 0 0 0 0 0 0 0 0 0|  Last+1 VMA page*
8065            ;*             +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                     *
8066            ;*                                                                   *
8067            ;*             +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                     *
8068            ;*    Case C   |1 1 1 1 1 1|0 0 0 0 0 0 0 0 0 0|     Fault         *
8069            ;*             +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                     *
8070            ;*                                                                   *
8071            ;*    When a PTE entry is case A, normal VMA operation proceeds.     *
8072            ;*    The PTE case B entry is indicated by a Page offset             *
8073            ;*    field of all zeros and a VMA suit of not all ones.             *
8074            ;*    In this case, the page is the last+1 VMA page which can        *
8075            ;*    be mapped, but not accessed.  The PTE case C entry is          *
8076            ;*    indicated by a Page offset of all zeros and a VMA suit         *
8077            ;*    of all ones.  In this case, the VMA page is not in memory      *
8078            ;*    and a page fault error is generated.                           *
8079            ;*                                                                   *
8080            ;*********************************************************************
```

```
LINE    ADDR    STATEMENT

8082            ;*********************************************************************
8083            ;*                                                                   *
8084            ;*        .LBPR - Load and map VMA pointer                           *
8085            ;*                                                                   *
8086            ;* - Calling Sequence:                                               *
8087            ;*                                                                   *
8088            ;*        JSB    .LBPR                                               *
8089            ;*        DEF    POINTER                                             *
8090            ;*                                                                   *
8091            ;* - Resolves pointer, double loads A and B and performs .LBP        *
8092            ;*                                                                   *
8093            ;*********************************************************************
8094            ;*****
8095            .LBPR:    CALL   SAVNLD     & AM2901 BR,TABQ,QREG,ADD,DZ
8096            /                           & CARRYL          ;
8097            /                           & LODUSR          ;
8098            /                           & LDMAR           ;
8099   006C6    /                           & DREAD           ; RESOLVE, READ POINTER; DL
8100                      JP     LPX05      & AM2901 PC,S5,RAMF,PASS,ZA
8101   006C7    /                           & SPNOP           ; SAVE PC
8102            ;*****
8103            ;*********************************************************************
8104            ;*                                                                   *
8105            ;*        .LPX - Add offset to VMA pointer and map                   *
8106            ;*                                                                   *
8107            ;* - Calling Sequence:                                               *
8108            ;*                                                                   *
8109            ;*        DLD    VMA_POINTER                                         *
8110            ;*        JSB    .LPX                                                *
8111            ;*        DEF    OFFSET                                              *
8112            ;*                                                                   *
8113            ;* - Resolve offset, double integer add to A & B, perform .LBP       *
8114            ;*                                                                   *
8115            ;*********************************************************************
8116            ;*****
8117            .LPX:     CALL   SAVEAB     & AM2901 BR,TABQ,QREG,ADD,DZ
8118            /                           & CARRYL          ;
8119            /                           & LODUSR          ;
8120            /                           & LDMAR           ;
8121   006C8    /                           & DREAD           ; RESOLVE,READ OFFSET ; NO
8122            LPXENT:   CALL   DADD       & AM2901 PC,S5,RAMF,PASS,ZA
8123   006C9    /                           & SPNOP           ; DOUBLE ADD OFFSET
8124            LPX05:    JP     VMAMAP     & AM2901 ,PC,RAMF,ADD,ZB
8125            /                           & CARRYH          ; INC PC PAST DEF
8126   006CA    /                           & LODMSR SWAPUM   ; USR = A
```

```
LINE    ADDR    STATEMENT

8128            ;*********************************************************************
8129            ;*                                                                   *
8130            ;*        .LPXR - Load VMA pointer, add offset and map               *
8131            ;*                                                                   *
8132            ;* - Calling Sequence:                                               *
8133            ;*                                                                   *
8134            ;*        JSB    .LPXR                                              *
8135            ;*        DEF    POINTER                                            *
8136            ;*        DEF    OFFSET                                             *
8137            ;*                                                                   *
8138            ;* - Resolve pointer, double load A & B, resolve offset,             *
8139            ;*   and double add offset.                                          *
8140            ;*                                                                   *
8141            ;* - Note:  OFFSET may NOT address A or B                            *
8142            ;*                                                                   *
8143            ;*********************************************************************
8144            ;*****
8145            .LPXR:    CALL  SAVNLD     & AM2901 BR,TABQ,QREG,ADD,DZ
8146            /                          & CARRYL         ;
8147            /                          & LODUSR         ;
8148            /                          & LDMAR          ;
8149   006CB    /                          & DREAD          ; RESOLVE,READ POINTER ; DL
8150                      CALL  READRES    & AM2901 PC,S5,RAMF,PASS,ZA
8151            /                          & LDMAR          ;
8152   006CC    /                          & CREAD          ; RESOLVE, READ OFFSET
8153                      CALL  DADD       & AM2901 ,PC,RAMF,ADD,ZB
8154   006CD    /                          & CARRYH         ; INC PC PAST 1ST DEF
8155                      JP    LPX05      & AM2901 ,,NOP,ADD,AQ
8156   006CE    /                          & SPNOP          ; FINISH
8157            ;*****
8158            ;*********************************************************************
8159            ;*                                                                   *
8160            ;*        .LBP - Map VMA pointer in A/B to logical address           *
8161            ;*                                                                   *
8162            ;* -At entry, A and B contain the VMA pointer.                       *
8163            ;*                                                                   *
8164            ;*********************************************************************
8165            ;*****
8166            .LBP0:    CALL  SAVEAB     & AM2901 PC,S5,RAMF,PASS,ZA
8167   006CF    /                          & LODUSR         ; SAVE PC,A,B
```

```
LINE    ADDR    STATEMENT

8169            ;*********************************************************************
8170            ;*                                                                   *
8171            ;*        VMAMAP - Perform VMA mapping function                      *
8172            ;*                                                                   *
8173            ;* - If A(15) is set, then this is a local reference.  The           *
8174            ;*   B reg is resolved and microcode exits.                          *
8175            ;*                                                                   *
8176            ;*********************************************************************
8177            ;*****
8178            VMAMAP:   CJP    VMAP01     & AM2901 ,B,QREG,PASS,ZB
8179            /                           & CONDUSR NSGN    ;
8180   006D0    /                           & LDMAR           ; Q=MAR=B
8181                      CCALL  INDRES     & AM2901 ,,NOP,ADD,AQ
8182            /                           & CONDUSR SGN     ;
8183   006D1    /                           & CMGO & DREAD    ;  RESOLVE FOR INDIRECTS
8184                      CONT              & AM2901 ,PC,NOP,SUBR,ZB
8185            /                           & CARRYH          ;
8186            /                           & LDMAR           ;
8187   006D2    /                           & IFETCH          ; FETCH NEXT INSTRUCTION
8188                      JZ                & AM2901 ,B,RAMF,PASS,ZQ
8189   006D3    /                           & SPNOP           ; B = RESOLVED ADDRESS
8190            ;
8191            VMAP01:   CONT              & AM2901 ,Y,RAMF,PASS,DZ
8192   006D4    /                           & IMM H#7800      ; Y = PAGE 30
8193                      CALL   GETPTE1    & AM2901 ,,NOP,AND,ZQ
8194            /                           & LODMSR RESET    ;
8195            /                           & ENBLO           ; PMAP FLAG = FALSE
8196   006D5    /                           & LDAER           ; AER -> SYSTEM MAP (0)
8197                      PUSH   H#005      & AM2901 A,X,RAMF,PASS,ZA
8198   006D6    /                           & SPNOP           ; X=A
8199                      RFCT              & AM2901 ,X,SRAMQL,PASS,ZB
8200   006D7    /                           & SHIFT B#0110    ;  SHIFT AB IN XQ LEFT 6
8201                      CALL   PTELKUP1   & AM2901 ,S1,NOP,PASS,ZB
8202   006D8    /                           & SPWR MAPWR      ; WRITE S1 TO MAP REG
8203                      CONT              & AM2901 ,Y,RAMF,PASS,DZ
8204            /                           & IMM H#7C00      ;
8205   006D9    /                           & LDMAR           ; Y=MAR=PAGE 31
8206                      CALL   PTELKUP    & AM2901 ,X,RAMF,ADD,ZB
8207   006DA    /                           & CARRYH          ; INC PAGEID
8208                      CONT              & AM2901 B,B,RAMF,NOTRS,DA
8209   006DB    /                           & IMM H#FC00      ; B = OFFSET
8210                      CONT              & AM2901 ,PC,NOP,SUBR,ZB
8211            /                           & CARRYH          ;
8212            /                           & LDMAR           ;
8213   006DC    /                           & IFETCH          ; FETCH NEXT INSTRUCTION
8214                      JZ                & AM2901 B,B,RAMF,OR,DA
8215   006DD    /                           & IMM H#7800      ; B = LOGICAL ADDRESS
```

```
LINE    ADDR    STATEMENT

8217            ;*********************************************************************
8218            ;*                                                                   *
8219            ;*        .PMAP - PAGE MAP                                           *
8220            ;*                                                                   *
8221            ;*  - Calling Sequence:                                              *
8222            ;*                                                                   *
8223            ;*        LDA   REG# IN MAPX                                         *
8224            ;*        LDB   PAGE_ID                                              *
8225            ;*        JSB   .PMAP                                                *
8226            ;*        <ERROR RETURN>                                             *
8227            ;*        <NORMAL RETURN>                                            *
8228            ;*                                                                   *
8229            ;*  - Used to load arbitrary map reg in MAPX for VMA                 *
8230            ;*                                                                   *
8231            ;*  - A reg contains page reg in MAPX to load.  B reg contains       *
8232            ;*    same VMA page_id as bits 25-10 of VMA pointer.                 *
8233            ;*                                                                   *
8234            ;*  - Perform VMA lookup as in .LBP.  If fault and A15 == 1          *
8235            ;*    then take <error return>.  If successful lookup, then          *
8236            ;*    load page reg in MAPX, inc A and B and take <normal ret>.      *
8237            ;*                                                                   *
8238            ;*  - 0 reg is set to 1                                              *
8239            ;*                                                                   *
8240            ;*  - E reg set if to 1 if last+1 page mapped                        *
8241            ;*                                                                   *
8242            ;*********************************************************************
8243            ;*****
8244            .PMAP:    CONT            & AM2901 ,Y,RAMF,PASS,DZ
8245   006DE    /                         & IMM H#7C00      ; Y -> PAGE 31
8246                      CALL  GETPTE    & AM2901 PC,S5,RAMF,PASS,ZA
8247            /                         & LODMSR SET      ;
8248   006DF    /                         & ENBLO           ; SET PMAP FLAG
8249                      CALL  PTELKUP   & AM2901 B,X,RAMF,PASS,ZA
8250   006E0    /                         & SPNOP           ; LOOK UP PTE ENTRY, MAP
8251                      CONT            & AM2901 A,S3,RAMA,PASS,DZ
8252   006E1    /                         & SPRD RL4        ; S3 = A LEFT 4
8253                      PUSH  H#001     & AM2901 S3,S3,RAMA,PASS,DZ
8254   006E2    /                         & SPRD RL4        ; S3 = A LEFT 8
8255                      RFCT            & AM2901 ,S3,SRAML,PASS,ZB
8256   006E3    /                         & SHIFT B#0010    ;  S3 = A LEFT 10
8257                      CALL  WRTMAP    & AM2901 ,S3,NOP,PASS,ZB
8258   006E4    /                         & LDMAR           ; MAR -> REG TO LOAD IN MAP
8259                      JP    INCBPC    & AM2901 ,A,RAMF,ADD,ZB
8260   006E5    /                         & CARRYH          ; INC A & B ; SKIP
```

```
LINE    ADDR    STATEMENT

8262            ;*********************************************************************
8263            ;*                                                                   *
8264            ;*        .IRES                                                      *
8265            ;*        Single integer subsript calculation                        *
8266            ;*                                                                   *
8267            ;*********************************************************************
8268            ;
8269            .IRES:    CALL  IRSUB      & AM2901 BR,TABQ,QREG,ADD,DZ
8270            /                          & CARRYL         ;
8271            /                          & LODUSR         ;
8272            /                          & LDMAR          ;
8273   006E6    /                          & DREAD          ; READ DOPE VECTOR
8274                      JZ               & AM2901 S6,PC,RAMA,ADD,ZA
8275            /                          & CARRYH         ;
8276            /                          & LDMAR          ;
8277   006E7    /                          & IFETCH         ; FETCH NEXT INST
8278            ;
8279            ;*********************************************************************
8280            ;*                                                                   *
8281            ;*        .IMAP                                                      *
8282            ;*        Do IRES, then map result                                   *
8283            ;*                                                                   *
8284            ;*********************************************************************
8285            ;
8286            .IMAP:    CALL  IRSUB      & AM2901 BR,TABQ,QREG,ADD,DZ
8287            /                          & CARRYL         ;
8288            /                          & LODUSR         ;
8289            /                          & LDMAR          ;
8290   006E8    /                          & DREAD          ; READ DOPE VECTOR
8291                      CONT             & AM2901 PC,S5,RAMF,PASS,ZA
8292   006E9    /                          & SPNOP          ; SAVE PC FOR VMAMAP
8293                      JP    VMAMAP     & AM2901 S6,PC,RAMF,ADD,ZA
8294            /                          & CARRYH         ; SET PC TO NEW VALUE, JP
8295   006EA    /                          & LODMSR SWAPUM  ; USR = MSR
```

```
LINE    ADDR    STATEMENT

8297            ;*********************************************************************
8298            ;*                                                                   *
8299            ;*        .JRES                                                      *
8300            ;*        Double integer subscript calculation                       *
8301            ;*                                                                   *
8302            ;*********************************************************************
8303            ;
8304            .JRES:    CALL  JRSUB      & AM2901 BR,TABQ,QREG,ADD,DZ
8305            /                          & CARRYL         ;
8306            /                          & LODUSR         ;
8307            /                          & LDMAR          ;
8308   006EB    /                          & DREAD          ; READ DOPE VECTOR
8309                      JZ               & AM2901 S6,PC,RAMA,ADD,ZA
8310            /                          & CARRYH         ;
8311            /                          & LDMAR          ;
8312   006EC    /                          & IFETCH         ;
8313            ;
8314            ;*********************************************************************
8315            ;*                                                                   *
8316            ;*        .JMAP                                                      *
8317            ;*        Do JRES, then map result                                   *
8318            ;*                                                                   *
8319            ;*********************************************************************
8320            ;
8321            .JMAP:    CALL  JRSUB      & AM2901 BR,TABQ,QREG,ADD,DZ
8322            /                          & CARRYL         ;
8323            /                          & LODUSR         ;
8324            /                          & LDMAR          ;
8325   006ED    /                          & DREAD          ;
8326                      CONT             & AM2901 PC,S5,RAMF,PASS,ZA
8327   006EE    /                          & SPNOP          ; SAVE PC FOR VMAMAP
8328                      JP    VMAMAP     & AM2901 S6,PC,RAMF,ADD,ZA
8329            /                          & CARRYH         ;
8330   006EF    /                          & LODMSR SWAPUM  ; USR = MSR
```

```
LINE    ADDR    STATEMENT

8332            ;*********************************************************************
8333            ;*                                                                   *
8334            ;*        VMA UTILITIES                                              *
8335            ;*                                                                   *
8336            ;*********************************************************************
8337            ;*****
8338            ;*********************************************************************
8339            ;*                                                                   *
8340            ;*        SAVEAB - SAVE A & B IN R6 & R7                             *
8341            ;*                 RESOLVE AND READ DEF                              *
8342            ;*********************************************************************
8343            ;
8344            SAVEAB:   CCALL READRES    & AM2901 B,S7,RAMF,PASS,ZA
8345   006F0    /                          & CONDUSR SGN     ; SAVE B; RESOLVE DEF
8346                      RET              & AM2901 A,S6,RAMF,PASS,ZA
8347   006F1    /                          & LODUSR          ; SAVE A; LODUSR FOR VMAMAP
8348            SAVNLD:   CCALL READRES    & AM2901 B,S7,RAMF,PASS,ZA
8349   006F2    /                          & CONDUSR SGN     ; SAVE B ; RESOLVE DEF
8350                      CONT             & AM2901 A,S6,RAMF,PASS,ZA
8351   006F3    /                          & SPNOP           ; SAVE A
8352            ;*********************************************************************
8353            ;*                                                                   *
8354            ;*        DLOAD - DOUBLE LOAD A & B FROM MEMORY                      *
8355            ;*                A := (QREG) , B := (QREG+1)                        *
8356            ;*                                                                   *
8357            ;*********************************************************************
8358            ;
8359            DLOAD:    CONT             & AM2901 ,,QREG,ADD,ZQ
8360            /                          & CARRYH          ;
8361            /                          & LDMAR           ;
8362   006F4    /                          & DREAD           ; READ SECOND WORD
8363            DLOAD1:   CONT             & AM2901 TAB,A,RAMF,PASS,DZ
8364   006F5    /                          & LODMSR          ; LOAD A ; MSR = A STATUS
8365                      RET              & AM2901 TAB,B,RAMF,PASS,DZ
8366   006F6    /                          & SPNOP           ; LOAD B, RETURN
8367            ;*********************************************************************
8368            ;*                                                                   *
8369            ;*        DADD - DOUBLE ADD (QREG,QREG+1) TO (A,B)                   *
8370            ;*                                                                   *
8371            ;*********************************************************************
8372            ;
8373            DADD:     CONT             & AM2901 ,,QREG,ADD,ZQ
8374            /                          & CARRYH          ;
8375            /                          & LDMAR           ;
8376   006F7    /                          & DREAD           ; READ SECOND WORD
8377            DADD1:    CONT             & AM2901 A,A,RAMF,ADD,DA
8378   006F8    /                          & CARRYL          ; ADD MSW TO A
8379                      CONT             & AM2901 B,B,RAMF,ADD,DA
8380            /                          & CARRYL          ;
8381   006F9    /                          & LODUSR          ; ADD LSW TO B
8382                      RET              & AM2901 ,A,RAMF,ADD,ZB
8383            /                          & CARRYREG        ;
8384   006FA    /                          & LODMSR CREG     ; PROPAGATE CARRY
```

```
LINE    ADDR    STATEMENT

8386            ;*********************************************************************
8387            ;*                                                                   *
8388            ;*        GETPTE - MAPS THE PTE PAGE INTO THE REGISTER               *
8389            ;*                 POINTED TO BY Y                                   *
8390            ;*                                                                   *
8391            ;*********************************************************************
8392            ;
8393            GETPTE:   CONT            & AM2901 ,,NOP,AND,ZQ
8394   006FB    /                         & LDAER          ; AER -> SYSTEM MAP (0)
8395            GETPTE1:  CONT            & AM2901 ,,NOP,PASS,DZ
8396            /                         & IMM VMAPTE     ;
8397            /                         & LDMAR          ;
8398   006FC    /                         & DREAD          ; READ PTE PAGE NO
8399                      CONT            & AM2901 MAPX,,NOP,PASS,DZ
8400   006FD    /                         & LDAER          ; RESTORE MAPX TO AER
8401                      CRET            & AM2901 Y,S1,RAMA,PASS,DZ
8402            /                         & CONDUSR NSGN   ; IF SIGN BIT SET, FAULT
8403            /                         & SEQFRZ         ; (VMA NOT INITIALIZED)
8404   006FE    /                         & LDMAR          ; S1=PTE PAGE NO;MAR=PTE AD
8405                      CONT            & AM2901 S1,,NOP,NOTRS,DA
8406            /                         & SPWR           ;
8407   006FF    /                         & IMM H#8000     ; SPWR IS MAPWR
8408                      CONT            & AM2901 Y,Y,RAMF,OR,DA
8409   00700    /                         & IMM H#8000     ; SET SIGN BIT OF Y FOR
8410            ;                                          ; PTE NOT INITIALIZED FAULT
```

```
LINE    ADDR    STATEMENT

8412            ;*********************************************************************
8413            ;*                                                                   *
8414            ;*        VMAFAULT                                                   *
8415            ;*                                                                   *
8416            ;* - VMA page fault handler.  This routine is common                 *
8417            ;*   to .LBP, .IMAP and .PMAP.  Note that .PMAP must                 *
8418            ;*   take an error return if A(15) is set.                           *
8419            ;*                                                                   *
8420            ;* - X reg is set to the PAGID of the page not in the PTE.           *
8421            ;*                                                                   *
8422            ;* - Y reg is set to the logical address of the PTE.                 *
8423            ;*                                                                   *
8424            ;* - Perform a JSB indirect thru location $VMA$ (VMALOC).            *
8425            ;*                                                                   *
8426            ;*********************************************************************
8427            ;*****
8428            ;*****    - Restore PC to value at entry to VMA
8429            ;*****    - Test for PMAP special handling
8430            ;*****
8431            VMAFAULT: CJP    VMAFPMAP   & AM2901 S5,PC,RAMF,SUBR,ZA
8432            /                           & CARRYH          ;
8433            /                           & CONDMSR OVR     ;
8434   00701    /                           & LDMAR           ; RESTORE PC; IF OVR, WAS P
8435                      CONT              & AM2901 S6,A,RAMF,PASS,ZA
8436   00702    /                           & SPNOP           ; RESTORE A
8437                      CJP    VMAFCDS    & AM2901 S7,B,RAMF,PASS,ZA
8438   00703    /                           & CONDEXT CSON    ; RESTORE B ; IF CSMODE ON,
8439            VMAF1:    CONT              & AM2901 ,,NOP,PASS,DZ
8440            /                           & IMM VMALOC      ;
8441            /                           & LDMAR           ;
8442   00704    /                           & CREAD           ; READ $VMA$ ADDRESS
8443                      CONT              & AM2901 ,,QREG,PASS,DZ
8444            /                           & LDMAR           ;
8445   00705    /                           & MWAIT           ; QREG = $VMA$; DO A PSEUDO
8446                      JP     JSBIVMA    & AM2901 PC,TAB,NOP,SUBR,ZA
8447            /                           & CARRYH          ;
8448            /                           & LDMDOR          ;
8449   00706    /                           & DWRITE          ;
8450            VMAFPMAP: CJP    VMAFCS0    & AM2901 ,A,NOP,PASS,ZB
8451            /                           & CONDEXT CSON    ;
8452   00707    /                           & LUSRCOND        ; TEST A FOR SIGN BIT
8453                      CJP    VMAF1      & AM2901 ,,NOP,ADD,AQ
8454   00708    /                           & CONDUSR NSGN    ; IF NOT SET, NORMAL FAULT
8455            VMAFP01:  JP     VMAFPAT    & AM2901 ,PC,RAMF,ADD,ZB
8456            /                           & CARRYH          ; INC PC
8457   00709    /                           & IFETCH          ; FETCH
8458            VMAFCS0:  CJP    VMAFP01    & AM2901 ,,NOP,ADD,AQ
8459   0070A    /                           & CONDUSR SGN     ; IF SIGN SET, ERROR RETURN
8460            VMAFCDS:  CONT              & AM2901 ,S7,RAMF,PASS,DZ
8461   0070B    /                           & IMM VMATRAP     ;
8462                      JP     FAULT      & AM2901 ,PC,RAMF,ADD,ZB
8463   0070C    /                           & CARRYH          ;
```

```
LINE    ADDR    STATEMENT

8465            ;*****************************************************************
8466            ;*                                                               *
8467            ;*        PTELKUP - Lookup page in PTE                           *
8468            ;*                                                               *
8469            ;* - At entry:                                                   *
8470            ;*                                                               *
8471            ;*        X   - Page ID                                          *
8472            ;*        R5 - PTE page number                                   *
8473            ;*    Y & MAR - Logical address of PTE (& map reg to update)     *
8474            ;*                                                               *
8475            ;* - At exit:                                                    *
8476            ;*                                                               *
8477            ;*        R15 - VMA Page loaded into map register                *
8478            ;*        E  - Set if last+1 page, Clear otherwise               *
8479            ;*        Map register updated                                   *
8480            ;*                                                               *
8481            ;*****************************************************************
8482            ;*****
8483            PTELKUP:  CONT            & AM2901 ,S1,NOP,PASS,ZB
8484   0070D    /                         & SPWR MAPWR      ; WRITE MAP REG
8485            PTELKUP1: CONT            & AM2901 X,S2,RAMF,NOTRS,DA
8486   0070E    /                         & IMM H#FC00      ; S2 = PTE OFFSET
8487                      CONT            & AM2901 Y,S2,NOP,OR,AB
8488            /                         & LDMAR           ;
8489   0070F    /                         & DREAD           ; READ PTE ENTRY
8490                      CONT            & AM2901 X,S2,RAMF,EXOR,AB
8491   00710    /                         & SPNOP           ; S2 = VMA SEG | 0
8492                      CONT            & AM2901 ,,QREG,PASS,DZ
8493            /                         & LODMSR RESET    ;
8494   00711    /                         & ENBLC           ; QREG = PTE ENT; MC = 0
8495            ;*****
8496            ;*****    - Determine which PTE entry case this is.
8497            ;*****
8498            ;*****    - Algorithm is:
8499            ;*****          1. Test for 1111110000000000B.  If same, then fault.
8500            ;*****          2. Test for VMA suit match. If not same then fault.
8501            ;*****          3. Test for page offset from PTE of zero.
8502            ;*****             If zero, then last+1 page case and map 177777Q
8503            ;*****                    else perform normal mapping
8504            ;*****
8505                      CONT             & AM2901 ,,NOP,EXOR,DQ
8506            /                          & LUSRCOND        ;
8507   00712    /                          & IMM H#FC00      ; TEST FOR NO ENTRY
8508                      CJP   VMAFAULT   & AM2901 S2,S2,RAMF,EXOR,AQ
8509   00713    /                          & CONDUSR Z       ; S2 = 0 | OFFSET ; JP IF N
8510            ;                                            ; (IF SUITS MATCH)
8511                      CONT             & AM2901 S2,,NOP,AND,DA
8512            /                          & LUSRCOND        ;
8513   00714    /                          & IMM H#FC00      ; MASK OFFSET, CHECK FOR SU
8514                      CJP   VMAFAULT   & AM2901 ,S2,NOP,PASS,ZB
8515   00715    /                          & CONDUSR NZ      ; IF NOT 0, FAULT
8516                      CCALL LSTPGP1    & AM2901 S1,S2,RAMF,ADD,AB
8517            /                          & CARRYL          ;
```

```
LINE    ADDR    STATEMENT

8518    00716   /                       & CONDUSR Z       ; IF OFFSET 0, LAST PAGE +
8519            WRTMAP:   RET           & AM2901 ,S2,NOP,PASS,ZB
8520    00717   /                       & SPWR MAPWR      ; WRITE MAP REG
8521            ;
8522            LSTPGP1:  RET           & AM2901 S2,S2,RAMF,SUBR,AB
8523            /                       & CARRYH          ;
8524            /                       & LODMSR SET      ; S2=FFFF, SET MC
8525    00718   /                       & ENBLC           ;
```

```
LINE    ADDR    STATEMENT

8527            ;*********************************************************************
8528            ;*                                                                   *
8529            ;*        IRSUB  -  Calculate subscripted array address              *
8530            ;*                                                                   *
8531            ;* -Calling Sequence:                                                *
8532            ;*                                                                   *
8533            ;*     JSB    .IRES/.IMAP                                            *
8534            ;*     DEF    DOPE VECTOR --------> DEC  N    # DIMENSIONS          *
8535            ;*     DEF    An   SUBSCRIPT N      DEC  Dn-1 DIMENSION N-1         *
8536            ;*     DEF    An-1 SUBSCRIPT N-1    DEC  Dn-2 DIMENSION N-2         *
8537            ;*       .                            .                            *
8538            ;*       .                            .                            *
8539            ;*       .                            .                            *
8540            ;*     DEF    A2   SUBSCRIPT 2      DEC  D1   DIMENSION 1           *
8541            ;*     DEF    A1   SUBSCRIPT 1      DEC  E    # WORDS PER ELEMENT   *
8542            ;*                                  OCT  UPPER HALF OF OFFSET       *
8543            ;*                                  OCT  LOWER HALF OF OFFSET       *
8544            ;*                                                                   *
8545            ;*  -Calculation for B(A1,A2,A3,A4) is:                              *
8546            ;*                                                                   *
8547            ;*     offset(B) + E * {A1 + D1*[A2 + D2*[A3 + D3*[A4 + 0]]]}        *
8548            ;*                                                                   *
8549            ;*                                                                   *
8550            ;*  -Notes:                                                          *
8551            ;*       - Subscripts are sign extended to 32 bits                   *
8552            ;*       - If a dimension is zero, it is really 2**16                *
8553            ;*       - Calculation is accumulated in A and B                     *
8554            ;*       - It is possible for # dimensions (N) to be zero            *
8555            ;*       - DEF'S can NOT be A/B addressable                          *
8556            ;*                                                                   *
8557            ;*********************************************************************
8558            ;*****
8559            IRSUB:    CCALL MRGIND     & AM2901 PC,S6,RAMF,PASS,ZA
8560   00719    /                          & CONDUSR SGN    ; RESOLVE DOPE VECTOR
8561                      CONT             & AM2901 PC,S4,RAMA,SUBS,DZ
8562            /                          & CARRYL         ;
8563            /                          & LODMSR         ;
8564   0071A    /                          & LDMAR          ; S4=MSR=N-1 ; MAR = 1ST SU
8565                      CCALL READRES    & AM2901 ,S1,RAMF,ADD,ZQ
8566            /                          & CARRYH         ; S1 = POINTER TO DOPE VECT
8567            /                          & CONDMSR NSGN   ; READ FIRST SUBSCRIPT
8568   0071B    /                          & CMG0 & CREAD   ; (IF THERE ARE ANY)
8569                      CJP   IDONE      & AM2901 S1,A,RAMA,AND,ZQ
8570            /                          & CONDMSR SGN    ; JP & READ IF DONE
8571            /                          & LDMAR          ; MAR = POINTER TO DOPE VEC
8572   0071C    /                          & CMG0 & DREAD   ; A = 0
8573                      CJP   INEG       & AM2901 ,B,RAMF,PASS,DZ
8574            /                          & CONDEXT MDIR15 ; B = FIRST SUBSCRIPT
8575   0071D    /                          & DREAD          ; SIGN EXTEND IF NEG
8576            ;*****
8577            ;*****    - Iteration loop
8578            ;*****    - Multiply accumulator by dimension
8579            ;*****    - Unsigned multiply.
```

```
LINE    ADDR    STATEMENT

8580            ;*****
8581            ;*****    - Algorithm is:
8582            ;*****                    S2 & Q := B * DIMENSION
8583            ;*****                         B := Q
8584            ;*****                    S2 & Q := A * DIMENSION + R13
8585            ;*****
8586            IR05:     LDCT  H#00F      & AM2901 B,S0,RAMF,PASS,ZA
8587   0071E    /                          & SPNOP          ; S0 = B
8588                      CJP   ZDIM       & AM2901 ,,QREG,PASS,DZ
8589            /                          & CONDUSR Z      ;
8590   0071F    /                          & SEQFRZ         ; QREG = DIMENSION ; TEST F
8591                      CPUSH            & AM2901 ,S2,SRAMQR,AND,ZB
8592            /                          & CONDUSR Z      ;
8593   00720    /                          & SHIFT B#0110   ; NEVER LOAD COUNTER
8594                      RFCT             & AM2901 MPY8,S2,SRAMQR,ADD,AB
8595            /                          & CARRYL         ;
8596   00721    /                          & SHIFT B#1011   ; UNSIGNED MULTIPLY
8597                      CONT             & AM2901 ,B,RAMF,PASS,ZQ
8598   00722    /                          & SPNOP          ; B=RESULT (DONE WITH B)
8599                      LDCT  H#00F      & AM2901 ,,QREG,PASS,DZ
8600   00723    /                          & LODUSR         ; QREG = DIMENSION
8601            ;*****
8602            ;*****  - Notes : - S2 is a partial product that must be added, UNSHIFT
8603            ;*****            - The following CPUSH never loads the counter
8604            ;*****
8605                      CPUSH            & AM2901 ,S3,SRAMQR,PASS,ZB
8606            /                          & CONDUSR Z      ;
8607   00724    /                          & SHIFT B#0110   ; SET Q0BUF
8608                      RFCT             & AM2901 MPY,S2,SRAMQR,ADD,AB
8609            /                          & CARRYL         ;
8610   00725    /                          & SHIFT B#1011   ; MULTIPLY BY A
8611            IR07:     LDCT  DADD1      & AM2901 ,S6,RAMF,ADD,ZB
8612            /                          & CARRYH         ;
8613            /                          & CONDMSR NZ     ;
8614            /                          & LDMAR          ;
8615   00726    /                          & CMGO & CREAD   ; READ NEXT DEF, IF NOT DON
8616                      CCALL READRES    & AM2901 ,A,RAMF,PASS,ZQ
8617   00727    /                          & CONDMSR NZ     ;
8618            IR08:     CJP   IDONE      & AM2901 ,S1,RAMF,ADD,ZB
8619            /                          & CARRYH         ;
8620            /                          & CONDMSR Z      ;
8621            /                          & LDMAR          ;
8622   00728    /                          & CMGO & DREAD   ; READ NEXT IN DOPE VECTOR
8623                      LDCT  IR05       & AM2901 ,S4,RAMF,SUBR,ZB
8624            /                          & CARRYH         ;
8625   00729    /                          & LODMSR         ; DECREMENT COUNT
8626                      CJP   INTERRPT   & AM2901 B,B,RAMF,ADD,DA
8627            /                          & CARRYL         ;
8628            /                          & LUSRCOND       ;
8629   0072A    /                          & CONDEXT INTRPT ; IF INT PEND, JP; ADD S TO
8630                      JRP   INEG       & AM2901 ,A,RAMF,ADD,ZB
8631            /                          & CARRYREG       ; PROPAGATE CARRY TO A
8632            /                          & CONDEXT MDIR15 ; IF SUBSCRIPT NEGATIVE,
8633   0072B    /                          & DREAD          ; SIGN EXTEND A.  READ DIM
```

```
LINE    ADDR    STATEMENT

8634            INEG:     JP    IR05       & AM2901 ,A,RAMF,SUBR,ZB
8635    0072C   /                          & CARRYH          ; ADD H#FFFF TO A
8636            ;
8637            ;*****
8638            ;*****    DIMENSION IS 0 - MEANS DIM IS REALLY 2^16
8639            ;*****
8640            ZDIM:     CONT               & AM2901 ,B,QREG,PASS,ZB
8641    0072D   /                          & SPNOP           ; QREG = B ;
8642                      CJP   IR07         & AM2901 ,B,RAMF,AND,ZQ
8643    0072E   /                          & CONDMSR NZ      ; B=0; IF ACTUAL DIM, DONE
8644                      JP    IR08         & AM2901 ,A,RAMF,AND,ZQ
8645    0072F   /                          & SPNOP           ; # WORDS/ELEMENT=0 -> A=0,
8646            ;
8647            IDONE:    JRP   DLOAD1       & AM2901 ,S1,RAMF,ADD,ZB
8648            /                          & CARRYH          ;
8649            /                          & CONDMSR SGN     ;
8650            /                          & LDMAR           ;
8651    00730   /                          & DREAD           ; LOAD OR ADD OFFSET TO (A,
```



```
LINE    ADDR    STATEMENT

8653            FDVPAT:   CONT               & AM2901 ,YEXP,SRAMR,ADD,ZB
8654            /                          & CARRYH          ; YEXP ENDS UP INCREMENTED
8655            /                          & CONDUSR NOVR    ; FOR COMPATIBILITY WITH RE
8656    00731   /                          & SHIFT B#0000    ;
8657                      JP    FDIVSUB      & AM2901 ,XEXP,RAMF,SUBR,ZB
8658    00732   /                          & CARRYH          ; CORRECT EXPONENT
8659                      JZ                 & AM2901 ,,QREG,ADD,AQ
8660    00733   /                          & SHIFT B#0110    ;
8661                      JZ                 & AM2901 ,,QREG,ADD,AQ
8662    00734   /                          & CARRYH          ;
8663    00735             FILLER
```

```
LINE    ADDR    STATEMENT

8665            ;*********************************************************************
8666            ;*                                                                   *
8667            ;*        JRSUB  -  Calculate subscripted array address for JRES     *
8668            ;*                                                                   *
8669            ;* -Calling Sequence:                                                *
8670            ;*                                                                   *
8671            ;*     JSB   .JRES/.JMAP                                             *
8672            ;*     DEF   DOPE VECTOR --------> DEC  N    # DIMENSIONS           *
8673            ;*     DEF   An   SUBSCRIPT N      DEC  Dn-1 DIMENSION N-1         *
8674            ;*     DEF   An-1 SUBSCRIPT N-1    DEC  Dn-2 DIMENSION N-2         *
8675            ;*      .                           .                                *
8676            ;*      .                           .                                *
8677            ;*      .                           .                                *
8678            ;*     DEF   A2   SUBSCRIPT 2      DEC  D1   DIMENSION 1           *
8679            ;*     DEF   A1   SUBSCRIPT 1      DEC  E    # WORDS PER ELEMENT    *
8680            ;*                                 OCT  UPPER HALF OF OFFSET        *
8681            ;*                                 OCT  LOWER HALF OF OFFSET        *
8682            ;*                                                                   *
8683            ;*  -Calculation for B(A1,A2,A3,A4) is:                              *
8684            ;*                                                                   *
8685            ;*     offset(B) + E * (A1 + D1*[A2 + D2*[A3 + D3*[A4 + 0]]])        *
8686            ;*                                                                   *
8687            ;*  - All subscripts and dimensions are double integers              *
8688            ;*                                                                   *
8689            ;*                                                                   *
8690            ;*  -Notes:                                                          *
8691            ;*      - Subscripts are sign extended to 32 bits                    *
8692            ;*      - If a dimension is zero, it is really 2**16                 *
8693            ;*      - Calculation is accumulated in A and B                      *
8694            ;*      - It is possible for # dimensions (N) to be zero             *
8695            ;*      - DEF'S can NOT be A/B addressable                           *
8696            ;*                                                                   *
8697            ;*********************************************************************
8698            ;*****
8699            JRSUB:    CCALL MRGIND     & AM2901 PC,S6,RAMF,ADD,ZA
8700            /                          & CARRYH          ; S6 = ^DEFS
8701   00736    /                          & CONDUSR SGN     ; RESOLVE DOPE VECTOR
8702                      CONT             & AM2901 PC,S4,RAMA,SUBS,DZ
8703            /                          & CARRYL          ; S4 = MSR = N-1
8704            /                          & LODMSR          ;
8705   00737    /                          & LDMAR           ; MAR = 1ST SUBSCR
8706                      CCALL READRES    & AM2901 ,S1,RAMF,ADD,ZQ
8707            /                          & CARRYH          ; S1 = POINTER TO DOPE VECT
8708            /                          & CONDMSR NSGN    ; READ FIRST SUBSCRIPT
8709   00738    /                          & CMGO & CREAD    ; (IF THERE ARE ANY)
8710                      CJP   IDONE      & AM2901 S1,S0,RAMA,ADD,ZQ
8711            /                          & CARRYH          ; S0 = 2ND WORD ADDRESS
8712            /                          & CONDMSR SGN     ; JP & READ IF DONE
8713            /                          & LDMAR           ; MAR = POINTER TO DOPE VEC
8714   00739    /                          & CMGO & DREAD    ; A = 0
8715                      CONT             & AM2901 S0,A,RAMA,PASS,DZ
8716            /                          & LDMAR           ; A = FIRST SUBSCRIPT UPPER
8717   0073A    /                          & DREAD           ; READ SECOND WORD
```

```
LINE    ADDR    STATEMENT

8718                    CONT            & AM2901 S1,S1,RAMA,ADD,ZB
8719            /                       & CARRYH        ; INC ^D.V.
8720    0073B   /                       & LDMAR         ;
8721                    CJP   JR08      & AM2901 ,B,RAMF,PASS,DZ
8722            /                       & CONDMSR Z     ;
8723    0073C   /                       & DREAD         ; READ DIM UPPER
8724            ;*****
8725            ;*****  - Iteration loop
8726            ;*****  - Multiply accumulator by dimension
8727            ;*****  - Unsigned multiply.
8728            ;*****
8729            ;*****  - Algorithm is:
8730            ;*****                  S2 & Q := B * DIMENSION
8731            ;*****                       B := Q
8732            ;*****                  S2 & Q := A * DIMENSION + R13
8733            ;*****
8734            JLOOP:  CONT            & AM2901 S1,S1,RAMA,ADD,ZB
8735            /                       & CARRYH        ; INC S1
8736    0073D   /                       & LDMAR         ;
8737                    CONT            & AM2901 ,S3,RAMF,OR,DZ
8738            /                       & CARRYH        ;
8739            /                       & DREAD         ;
8740    0073E   /                       & LODUSR        ;
8741                    CJP   JSWP      & AM2901 B,S0,RAMF,PASS,ZA
8742    0073F   /                       & CONDUSR UGT   ; JP IF NOT ZEROS OR ONES
8743            JENT0:  CONT            & AM2901 ,S5,RAMF,PASS,DZ
8744    00740   /                       & SPNOP         ; S5 = DIM LOWER
8745            JENT:   LDCT  H#00F     & AM2901 ,S5,QREG,PASS,ZB
8746    00741   /                       & SPNOP         ; QREG = S5
8747                    CPUSH           & AM2901 ,S2,SRAMQR,AND,ZB
8748            /                       & CONDMSR SGN   ;
8749    00742   /                       & SHIFT B#0110  ; NEVER LOAD COUNTER
8750                    RFCT            & AM2901 MPY8,S2,SRAMQR,ADD,AB
8751            /                       & CARRYL        ;
8752    00743   /                       & SHIFT B#1011  ; UNSIGNED MULTIPLY
8753                    CONT            & AM2901 ,B,RAMF,PASS,ZQ
8754    00744   /                       & SPNOP         ; B=RESULT (DONE WITH B)
8755                    LDCT  H#00F     & AM2901 ,S5,QREG,PASS,ZB
8756    00745   /                       & SPNOP         ; QREG = DIMENSION
8757            ;*****
8758            ;*****  - Notes : - S2 is a partial product that must be added, UNSHIFT
8759            ;*****            - The following CPUSH never loads the counter
8760            ;*****
8761                    CPUSH           & AM2901 ,S7,SRAMQR,AND,ZB
8762            /                       & CONDMSR SGN   ;
8763    00746   /                       & SHIFT B#0110  ; SET Q0BUF
8764                    RFCT            & AM2901 MPY,S2,SRAMQR,ADD,AB
8765            /                       & CARRYL        ;
8766    00747   /                       & SHIFT B#1011  ; MULTIPLY BY A
8767                    LDCT  DADD1     & AM2901 ,S3,NOP,PASS,ZB
8768    00748   /                       & LODUSR        ; USR = DIM UPPER SIGN
8769                    CJP   JR05      & AM2901 ,A,RAMF,PASS,ZQ
8770    00749   /                       & CONDUSR NSGN  ;
8771                    CONT            & AM2901 S5,A,RAMF,SUBR,AB
```

```
LINE   ADDR   STATEMENT

8772   0074A  /                        & CARRYL          ; SUBTRACT DIM LOWER FROM A
8773          ;                                          ; !NO CHECK FOR OVERFLOW!
8774          JR05:    CJP   IDONE     & AM2901 ,S1,NOP,PASS,ZB
8775          /                        & CONDMSR Z       ;
8776          /                        & LDMAR           ;
8777   0074B  /                        & CMGO & DREAD    ; READ NEXT IN DOPE VECTOR
8778                   CALL  READRES   & AM2901 S6,S6,RAMA,ADD,ZB
8779          /                        & CARRYH          ;
8780          /                        & LDMAR           ;
8781   0074C  /                        & CREAD           ; READ, RESOLVE NEXT DEF
8782                   CONT            & AM2901 ,,NOP,ADD,ZQ
8783          /                        & CARRYH          ;
8784   0074D  /                        & LDMAR           ; MAR = 2ND WORD ADDRESS
8785                   CONT            & AM2901 ,S4,RAMF,SUBR,ZB
8786          /                        & CARRYH          ;
8787          /                        & LODMSR          ;
8788   0074E  /                        & DREAD           ;
8789                   CONT            & AM2901 A,A,RAMF,ADD,DA
8790   0074F  /                        & CARRYL          ;
8791                   CONT            & AM2901 B,B,RAMF,ADD,DA
8792          /                        & CARRYL          ;
8793   00750  /                        & LODUSR          ;
8794                   CJP   INTERRPT  & AM2901 ,A,RAMF,ADD,ZB
8795          /                        & CARRYREG        ; PROPAGATE CARRY
8796   00751  /                        & CONDEXT INTRPT  ;
8797                   CJP   JLOOP     & AM2901 S1,S1,RAMA,ADD,ZB
8798          /                        & CARRYH          ;
8799          /                        & CONDMSR NZ      ;
8800          /                        & LDMAR           ; READ DIMENSION UPPER
8801   00752  /                        & DREAD           ;
8802          JR08:    CONT            & AM2901 ,S3,RAMF,AND,ZB
8803   00753  /                        & SPNOP           ; S3 = 0
8804                   JP    JENT0     & AM2901 B,S0,RAMF,PASS,ZA
8805   00754  /                        & SPNOP           ; LAST TIME THROUGH LOOP
8806          ;
8807          JSWP:    CONT            & AM2901 B,S5,RAMF,PASS,ZA
8808   00755  /                        & SPNOP           ;
8809                   CONT            & AM2901 A,S0,RAMF,PASS,ZA
8810   00756  /                        & SPNOP           ;
8811                   CONT            & AM2901 S3,A,RAMF,PASS,ZA
8812   00757  /                        & SPNOP           ;
8813                   CONT            & AM2901 S0,S3,RAMF,PASS,ZA
8814   00758  /                        & SPNOP           ;
8815                   JP    JENT      & AM2901 ,S0,RAMF,PASS,DZ
8816   00759  /                        & SPNOP           ;
```

```
LINE    ADDR    STATEMENT

8818            ;
8819            ;       VCP ROM CHECKSUM CHECK
8820            ;
8821            CHECKSUM: CONT          & AM2901 ,S1,SRAML,PASS,DZ
8822            /                       & IMM H#0020      ; AER POINTS TO BOOT MEMORY
8823    0075A   /                       & LDAER           ; CLEAR E,S1=H#0080
8824                      CONT          & AM2901 ,X,RAMF,PASS,DZ
8825            /                       & IMM H#2000      ; FIRST ROM LOCATION
8826            /                       & LDMAR           ; X = ADDRESS POINTER
8827    0075B   /                       & DREAD           ; READ IT
8828                      CONT          & AM2901 ,S0,RAMF,PASS,DZ
8829    0075C   /                       & IMM H#FF00      ; S0 = BYTE MASK
8830                      CONT          & AM2901 ,Y,RAMF,PASS,DZ
8831    0075D   /                       & SPNOP           ;
8832                      CONT          & AM2901 Y,S0,RAMF,AND,AB
8833    0075E   /                       & SPNOP           ; ACCUMULATE CHECKSUM IN Y
8834                      CONT          & AM2901 S0,,NOP,SUBS,DA
8835            /                       & CARRYH          ; COMPARE ROM SIZE
8836            /                       & IMM H#0800      ; COMPARE TO 8
8837    0075F   /                       & LUSRCOND        ;
8838                      CCALL EIGHTK  & AM2901 ,,NOP,ADD,AQ
8839    00760   /                       & CONDUSR Z       ; IF ZERO, 8K
8840                      PUSH  H#FFE   & AM2901 ,X,RAMF,ADD,ZB
8841            /                       & CARRYH          ; INC X
8842            /                       & LDMAR           ;
8843    00761   /                       & DREAD           ; READ NEXT WORD
8844                      CONT          & AM2901 ,X,RAMF,ADD,ZB
8845            /                       & CARRYH          ; INC ADDRESS
8846    00762   /                       & LDMAR           ;
8847                      RFCT          & AM2901 Y,Y,RAMF,ADD,DA
8848            /                       & CARRYL          ; ADD TO CHECKSUM
8849            /                       & LODUSR          ;
8850    00763   /                       & DREAD           ; READ NEXT WORD
8851                      CRET          & AM2901 ,,NOP,ADD,AQ
8852    00764   /                       & CONDUSR Z       ; IF Z, CHECKSUM OK
8853                      JP    $       & AM2901 ,S1,NOP,PASS,ZB
8854    00765   /                       & SPWR LEDWR      ; WRITE TO LEDS
8855            EIGHTK:   PUSH  H#FFE   & AM2901 ,X,RAMF,ADD,ZB
8856            /                       & CARRYH          ;
8857            /                       & LDMAR           ;
8858    00766   /                       & DREAD           ;
8859                      CONT          & AM2901 ,X,RAMF,ADD,ZB
8860            /                       & CARRYH          ;
8861    00767   /                       & LDMAR           ;
8862                      RFCT          & AM2901 Y,Y,RAMF,ADD,DA
8863            /                       & CARRYL          ;
8864    00768   /                       & DREAD           ;
8865                      RET           & AM2901 Y,Y,RAMF,ADD,DA
8866    00769   /                       & MWAIT           ;
```

```
A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]     1:35 PM  TUE., 13  SEPT, 1983 PAGE 226
LINE    ADDR    STATEMENT

8868            ;
8869            ;          PATCH AREA  (AND VECTOR AREA FOR 4Kx8 BASESET PROMS)
8870            ;
8871            SETAB:     CONT             & AM2901 MAPX,S0,RAMF,PASS,DZ
8872   0076A    /                           & SPNOP          ; S0 = MAPX
8873                       RET              & AM2901 S0,MAPX,NOP,OR,DA
8874            /                           & IMM H#0040     ; OR IN ABREF BIT
8875   0076B    /                           & LDAER          ; AER = MAPX
A600+ BASESET MICROCODE 6/15/83 ** [&LBRB1]     1:35 PM  TUE., 13  SEPT, 1983 PAGE 227
LINE    ADDR    STATEMENT

8877            ;*************************************************************************
8878            ;*                                                                       *
8879            ;*         105XXX INSTRUCTIONS UIT TO 7FF                                *
8880            ;*                                                                       *
8881            ;*************************************************************************
8882   007FF                ORG H#7FF
8883   007FF    ;*****
8884            UIT105:    JP    UIT        & AM2901 ,,NOP,ADD,AQ
8885   007FF    /                           & SPNOP          ;
8886            ;
8887            END


  TOTAL ASSEMBLY ERRORS =     0
```

SYMBOL TABLE

A	A	00000	AB	A	00001	ABFETCH	D		ADD	A	00000
ADQ.	A	00407	ADX.	A	0018A	AD.	A	00003	AD.I	A	00002
ALIGN	A	004B8	AM2901	D		AND	A	00004	ANDI	A	00004
AND.	A	00005	AQ	A	00000	ASCONT	A	00043	ASGEOP	A	00042
ASGSP	A	0000C	ASL	A	000C0	ASLOVFL	A	0012E	ASR	A	000D2
B	A	00001	BR	A	00017	BREL	A	00016	BYTESWAP	A	001BF
C	A	0000B	CAB	A	00010	CALL	D		CARRYEXT	D	
CARRYH	D		CARRYL	D		CARRYNUC	D		CARRYREG	D	
CARRYUC	D		CARYUCON	D		CBS	A	001A3	CBT	A	001C1
CBTDONE	A	001CF	CCALL	D		CCQ.	A	00403	CC.	A	0003F
CHECK	A	00438	CHECKSUM	A	0075A	CI	A	00008	CIQ.	A	00408
CIR	A	0001F	CJP	D		CJPP	D		CLC00	A	00051
CLC04	A	00088	CLC05	A	00093	CLC05C	A	00092	CLC06	A	000A8
CLD	D		CLF00	A	00052	CLF02	A	00071	CLF05	A	00094
CLF06	A	000A9	CLO	A	00061	CLRCS	A	00007	CLRDTST	A	0000A
CLRF0	A	00053	CLRF5	A	00095	CLRF6	A	000AA	CLRMPEN	A	00000
CLRMPI	A	00004	CLRO	A	00062	CLRPEI	A	00009	CLRPFWI	A	0000E
CLRPSFF	A	00002	CLRTBT	A	0000C	CLRTDI	A	00006	CL.	A	00040
CMGO	D		CMSGN	D		CMW	A	001EA	CMWNEQ	A	001F1
CM.	A	00041	CNMGO	D		CON12	D		CONDEXT	D	
CONDMSR	D		CONDUSR	D		CONT	D		COPYABXY	A	0018E
COPYARGS	A	00455	COPYXYAB	A	0018F	CORZ	A	00009	CPUID	A	00005
CPUSH	D		CP.	A	00007	CP.I	A	00006	CREAD	D	
CREG	A	0000E	CRET	D		CSOFF	A	0000C	CSON	A	0000D
CVECT	D		CWDCODE	A	0001B	CXY	A	00012	CZ.	A	00406
C.CQ	A	00400	C.CQ05	A	00402	C.Z	A	00405	DA	A	00005
DADD	A	006F7	DADD1	A	006F8	DBLADD	A	005A7	DBLARG1	A	00507
DBLARG1A	A	00509	DBLARG2	A	00511	DBLARG2A	A	00513	DBLARG2B	A	0051A
DBLIE	A	00208	DBLILSW	A	0000F	DBLIMSW	A	0000B	DCZFER	A	00284
DDI01	A	00242	DDI02	A	00245	DDI03	A	00249	DDI04	A	00254
DDI05	A	0025C	DDI06	A	0025F	DDI08	A	00263	DFIXOFL	A	005AC
DIARG	A	00269	DIV	A	0001B	DIV05	A	0010A	DIV20	A	00114
DIV25	A	00118	DIV30	A	0011A	DIV40	A	0011B	DIVCOND	D	
DIVCONT	A	00107	DIVD	A	000DC	DIVOVFL	A	0011D	DIVUCOND	D	
DLD	A	000DE	DLOAD	A	006F4	DLOAD1	A	006F5	DMP1	A	00224
DMP2	A	00229	DMP3	A	00234	DMP9	A	00237	DMSEXIT	A	00385
DNCOUNT	A	0060F	DNCOUNT1	A	0060E	DNCOUNT2	A	0060D	DNEGATE	A	00569
DNEGATE1	A	0056F	DNEGATE4	A	00678	DNEGATE8	A	0067C	DNORM	A	005B2
DNORM1	A	005B3	DNORM12	A	005D5	DNORM13	A	005DB	DNORM14	A	005DD
DNORM15	A	005E1	DNORM2	A	005B9	DNORM3	A	005BD	DNORM6	A	005C0
DNORM7	A	005C1	DNORM8	A	005C4	DOFL	A	00615	DOFLUFL	A	00613
DPACK	A	005F0	DPACK1	A	005F1	DQ	A	00006	DREAD	D	
DROUND	A	005E5	DROUND1	A	005ED	DST	A	000E2	DSTORE	A	005F8
DSTORE1	A	005F9	DSXY	A	00190	DWRITE	D		DZ	A	00007
DZERO	A	0060E	ECIRRD	A	00005	EFAULT	A	0041D	EIGHTK	A	00766
ENBLC	D		ENBLO	D		ENCN	D		ENDCBT	A	001F7
ENDCMW	A	001F5	ENDEV	A	003A5	ENDOD	A	003BE	ENE	A	0000D
ENECI	A	00009	ENVE	A	0000F	EVBYTE	A	001BE	EVENT	A	003A2
EVLOOP	A	003A1	EVTOOD	A	003B1	EXIT0	A	08B0F	EXIT1	A	08B0D
EXIT2	A	08B0E	EXNOR	A	00007	EXOR	A	00006	EXSKP	A	0043E
FADD1	A	004AB	FADD2	A	004BA	FAULT	A	0041E	FDIV1	A	004E3

FDIV2	A	004E8	FDIV3	A	004ED	FDIV5	A	004F9	FDIVSUB	A	004FA
FDVPAT	A	00731	FETCH	A	00000	FILL	A	003E0	FILLCONT	A	003E8
FILLER	D		FILLOOP	A	003E9	FILLTOOD	A	003EC	FIXD1	A	00491
FIXDONE	A	00486	FIXMAR	A	0004E	FIXOVR	A	00489	FIXPC	A	000ED
FIXZERO	A	00488	FLT	A	0048A	FLTD1	A	00496	FLTD2	A	00498
FLTD3	A	0049B	FLUN1	A	002B7	FMOD	A	003D8	FMPYSUB	A	004D4
FOFL	A	002DE	FOFLUFL	A	002DC	FP	A	0000E	FROMOD	A	003C2
GENMPV	A	00106	GETCQ	A	00359	GETPTE	A	006FB	GETPTE1	A	006FC
HLT	A	00045	HLT1	A	00048	ICRS	A	00008	IDONE	A	00730
IFETCH	D		IMAPLOC	A	00041	IMM	D		IMMB	D	
INCB	A	001B6	INCBPC	A	003B0	INCEXP	A	002C9	INCFTCH	A	0000F
INCPC	A	00442	INCQ	A	000EC	INCS5	A	003F9	INDMS	A	00394
INDMS1	A	00395	INDRES	A	00129	INDRES1	A	0012A	INDRES2	A	0012C
INEG	A	0072C	INIT	A	0000C	INITIAL	A	001F8	INTCBT	A	001D0
INTCMW	A	001FE	INTD	A	00277	INTDEAD	A	00150	INTERRPT	A	00102
INTEV	A	003DC	INTFTCH	A	0015F	INTIO	A	00168	INTMVW	A	0038E
INTOD	A	003DA	INTPARTY	A	00162	INTPEND	A	00200	INTPFW	A	00173
INTPON	A	0014C	INTPONOK	A	00152	INTPROT	A	00170	INTRPT	A	0000A
INTSLRQ	A	0016B	INTTBG	A	00165	INTTBL	A	000F0	INVERT	A	00005
IODC0001	A	0013D	IODC0010	A	0013F	IODC0011	A	00141	IODC0110	A	00143
IODC0111	A	00145	IODC1001	A	00146	IODC1010	A	00149	IODC1011	A	0014A
IODC1110	A	0014B	IODCXXXX	A	0013C	IOGGE20	A	0004A	IOGLT20	A	0004F
IOHSHAK0	A	00137	IOHSHAK1	A	00139	IOHSHAK2	A	0013A	IOHSHAKE	A	00135
IOR	A	00009	IORD	A	00004	IORI	A	00008	IORQ	A	00001
IOWR	A	0000C	IQLOC	A	00042	IR05	A	0071E	IR07	A	00726
IR08	A	00728	IR0T3	A	00008	IR11	A	00004	IR2	A	0000F
IRSKBF	A	00003	IRSP	A	00006	IRSUB	A	00719	ISXY	A	00191
ISZ	A	0000B	ISZI	A	0000A	JENT	A	00741	JENT0	A	00740
JLOOP	A	0073D	JLY	A	00192	JL.	A	000C8	JMAP	D	
JMP	A	00010	JMPI	A	0000D	JP	D		JPP	D	
JPTDI	A	0000E	JPY	A	00194	JR05	A	0074B	JR08	A	00753
JRP	D		JRSUB	A	00736	JSB	A	00014	JSBI	A	00011
JSBIVMA	A	00013	JSRP	D		JSWP	A	00755	JTAB	D	
JZ	D		JZA	A	0012F	L4MORE	A	001C0	LABXY	A	00196
LASTWD	A	003FF	LBT	A	001B1	LC05	A	001CC	LCMW	A	001EC
LDAER	D		LDCT	D		LDCTY	A	00003	LDIM1	A	00002
LDMAPD	A	00000	LDMAR	D		LDMDOR	D		LDMP	A	0032C
LDMP5	A	00330	LDMP7	A	00334	LDST	D		LDX.	A	00199
LD.	A	00017	LD.I	A	00016	LEDWR	A	00001	LI05A	A	0009C
LI.00C	A	00054	LI.01	A	00064	LI.01C	A	00063	LI.02C	A	00074
LI.04	A	00089	LI.05C	A	00098	LI.05H	A	00096	LI.06	A	000AC
LI.06C	A	000AB	LI.07	A	000BE	LMOVBYT	A	00384	LMOVBYT1	A	00386
LMVW	A	001E6	LODMSR	D		LODUSR	D		LODUSRCI	D	
LODUSROC	D		LODUSROR	D		LOOP	D		LOWSC	A	00004
LPMR0	A	00322	LPX05	A	006CA	LPXENT	A	006C9	LSHIFT	A	0059D
LSL	A	000D6	LSR	A	000D9	LSTPGP1	A	00718	LT20ENT	A	0004B
LUSRCOND	D		LUSRSP	D		LWD1	A	0031C	LWD2	A	0031F
MAPD1	A	0001D	MAPD2	A	0001E	MAPIT	A	00469	MAPWR	A	00008
MAPX	A	0001C	MBT	A	001D1	MBYTE	A	0039A	MDIR15	A	00000
MEVSAVE	A	0038C	MEVTOEV	A	0038F	MFMOD	A	00390	MIAK	A	00002
MICREVID	A	00003	MI.01	A	00067	MI.01C	A	00066	MI.04	A	0008A
MKLROFF	A	00005	MKLRON	A	0000D	MODSAVE	A	00392	MODTOEV	A	00391
MODTOOD	A	00393	MOV00	A	0035F	MOV01	A	00361	MOV02	A	00363
MOV10	A	00365	MOV11	A	00367	MOV12	A	00369	MOV20	A	0036B

MOV21	A	0036D	MOV22	A	0036F	MOVBYTA	A	0037A	MOVBYTAB	A	0037D
MOVBYTB	A	0037F	MOVEBYTE	A	00381	MOVWORD	A	00374	MOVWRDA	A	00371
MOVWRDAB	A	00372	MOVWRDB	A	00373	MP	A	00005	MPEN	A	00007
MPLOWSC	A	00104	MPY	A	00013	MPY0	A	000E5	MPY8	A	0001B
MRGIND	A	00124	MRGIND2	A	00125	MVW	A	001E5	MVW1	A	00376
MVWLOOP	A	00375	MWAIT	D		MWRBYTE	A	0038D	NARG	A	00000
NC	A	0000A	NCNZ	A	00008	NGL01	A	002FD	NGL02	A	002FE
NINTRP	A	0000B	NMPEN	A	00008	NOABIN	A	00009	NOARGS	A	00447
NOP	A	00001	NORM	A	002BB	NORM1	A	002C2	NORMEND	A	002CA
NOSIGNS	A	002C7	NOTRS	A	00005	NOVR	A	00006	NSGN	A	0000E
NSIGN	A	00006	NZ	A	00004	ODBYTE	A	001BD	ODENT	A	003B9
ODLOOP	A	003B7	ODTOOD	A	003C5	ONEBYTE	A	003CE	ONE.A	A	003D3
ONE.B	A	003D6	OR	A	00003	OT.00	A	00057	OT.00C	A	00056
OT.01	A	0006B	OT.01C	A	0006A	OT.02C	A	00076	OT.02H	A	00077
OT.04	A	0008C	OT.06	A	000AF	OT.06C	A	000AE	OVR	A	00007
P1WR	A	00000	P2WR	A	00004	PASS	A	00003	PAT02C	A	000EE
PC	A	00007	PCA00	A	00454	PCA05	A	00458	PCALMPR	A	00475
PCALMPV	A	00476	PCASUB1	A	00449	PCASUB2	A	00451	PCAXSUB	A	00462
PDIRRD	A	00008	PDONE	A	00418	PELENH	A	00003	PELENL	A	00002
PINDR	A	00459	PORM	A	00014	PORQ	A	00015	PORTRD	A	00000
POSIMAPR	A	00328	PRLEN	A	00001	PTELKUP	A	0070D	PTELKUP1	A	0070E
PUSH	D		PWR2POS	A	002E8	Q	A	00006	QREG	A	00000
RAMA	A	00002	RAMF	A	00003	RDCST	A	0045D	RDRES2	A	00121
READIND	A	00123	READRES	A	0011E	RESET	A	00003	RET	D	
RFCT	D		RFETCH	D		RL4	A	00004	ROTATE	A	0000A
ROTATEC	A	00009	ROUND	A	002CB	ROUND0	A	002CF	ROUND1	A	002D2
ROUND2	A	002D5	RPCT	D		RRL	A	000CA	RRR	A	000CC
RSTRE	A	001F3	S0	A	00008	S1	A	00009	S2	A	0000A
S3	A	0000B	S4	A	0000C	S5	A	0000D	S6	A	0000E
S7	A	0000F	SABXY	A	0019B	SAMEMAP	A	00378	SAVEAB	A	006F0
SAVEIMAP	A	0017F	SAVEINT	A	0017A	SAVNLD	A	006F2	SBS	A	001A7
SBT	A	001B7	SDSPL	A	00411	SEGTRAP	A	0000B	SEQFRZ	D	
SET	A	00001	SETAB	A	0076A	SETCS	A	0000F	SETDTST	A	0000B
SETMAPS	A	00177	SETMPEN	A	00001	SETMPI	A	00005	SETMSR	A	003F8
SETPINT	A	0027B	SETPL	A	00272	SETPSFF	A	00003	SETTBT	A	0000D
SETTDI	A	00007	SFB	A	001D4	SFC00	A	00058	SFC00.C	A	0005A
SFC02	A	0007D	SFC02C	A	0007B	SFC04	A	0008D	SFC05	A	0009F
SFC05C	A	0009D	SFC06	A	000B5	SFC06C	A	000B3	SFS00	A	0005C
SFS00.C	A	0005E	SFS02	A	00081	SFS02C	A	0007F	SFS04	A	0008F
SFS05	A	000A3	SFS05C	A	000A1	SFS06	A	000B9	SFS06C	A	000B7
SGE	A	00002	SGLD	A	0047A	SGN	A	0000F	SGT	A	00000
SHIFT	D		SHIFT1	A	0058D	SIGN	A	00005	SIGNEXT	A	00618
SIMP0	A	00344	SINTRQ	A	00002	SKIFNZ	A	0005D	SKIFZ	A	00059
SKIP	A	0015E	SKPEX	A	00444	SLACK	A	00001	SLE	A	00001
SLOOP	A	001DB	SLT	A	00003	SL.CONT	A	0002E	SOC.C	A	0006C
SOC.H	A	0006D	SODD	A	001E0	SOS.C	A	0006E	SOS.H	A	0006F
SPETC	D		SPIND	D		SPMR0	A	00325	SPNOP	D	
SPRD	D		SPWR	D		SR1CLE	A	0002C	SR1CLESL	A	0002F
SR1SL.	A	0002D	SRAML	A	00007	SRAMQL	A	00006	SRAMQR	A	00004
SRAMR	A	00005	SRG0EL.	A	00025	SRG0EL.D	A	00026	SRG0ER.	A	00027
SRG0ER.D	A	00028	SRG0NOP	A	00029	SRG0R.L	A	0002A	SRG0R.R	A	0002B
SRG0.LF	A	0001D	SRG0.LR	A	0001E	SRG0.LS	A	00020	SRG0.RS	A	00023
SRG1	A	00009	SRG2	A	0000A	SRG2EL.	A	00038	SRG2EL.D	A	00039
SRG2ER.	A	0003A	SRG2ER.D	A	0003B	SRG2NOP	A	0003C	SRG2R.L	A	0003D

SRG2R.R	A	0003E	SRG2.LF	A	00030	SRG2.LR	A	00031	SRG2.LS	A	00033
SRG2.RS	A	00036	STC02	A	00083	STC04	A	00091	STC05	A	000A6
STC05C	A	000A5	STC06	A	000BC	STC06C	A	000BB	STC07	A	000BF
STEV	A	003AC	STF00	A	00060	STF02	A	00085	STF05	A	000A7
STF06	A	000BD	STMP	A	00336	STMP5	A	0033A	STMP7	A	0033F
ST0	A	00070	STRD	A	00007	STWMAP	A	00180	STX.	A	0019E
ST.	A	00019	ST.I	A	00018	SUBR	A	00001	SUBS	A	00002
SVWMAP	A	00189	SWAPE0	A	00004	SWAPFARG	A	004C1	SWAPUM	A	00002
SWMP0	A	00340	SWPDBL	A	0061A	SWRD	A	00006	TAB	A	00011
TABQ	A	00013	TADD0	A	00575	TADD1	A	00579	TADD11	A	00597
TADD2	A	0057C	TBS	A	001AB	TDIV1	A	0069C	TDIV2	A	006A0
TDIV3	A	006AC	TDIV4	A	006AF	TDIV5	A	006BC	TDIV6	A	006BE
TDIV9	A	006C4	TDIVSKIP	X	00947	TEND	A	00521	TFTD1	A	0055F
TFTS1	A	0052F	TFTS2	A	00531	TFTS3	A	00536	TFXD1	A	0053E
TFXD2	A	00544	TFXD3	A	00545	TFXD4	A	00546	TFXD5	A	0054C
TFXD7	A	00552	TFXD8	A	00558	TFXS1	A	00522	TFXS2	A	00526
TFXS3	A	00528	TFXS0V	A	0052B	TMPYEX1	A	00672	TMPYEX2	A	00677
TMPYEXIT	A	0066E	TOOD	A	003D9	TSHIFT	A	0058C	TWB	D	
TWIEXIT	A	00333	UCDIVCND	D		UGT	A	0000C	UIT	A	000FF
UIT105	A	007FF	ULE	A	0000D	UNPACK	A	004FD	UNPACK1	A	00501
UNPACK2	A	00506	VANILLA	A	0000A	VECT	D		VMAF1	A	00704
VMAFAULT	A	00701	VMAFCDS	A	0070B	VMAFCS0	A	0070A	VMAFP01	A	00709
VMAFPAT	A	003FA	VMAFPMAP	A	00707	VMALOC	A	00004	VMAMAP	A	006D0
VMAP01	A	006D4	VMAPTE	A	00002	VMATRAP	A	0000A	WINDR	A	002A1
WORDCNT	A	0000B	WRD1	A	0058A	WRD2	A	00588	WREXITS	A	00431
WRITEIQ	A	0017E	WRTMAP	A	00717	X	A	00002	XABXY	A	001A0
XCCONT	A	00311	XC.10	A	0030F	XC.20	A	00314	XEXP	A	00009
XFCONT	A	0028A	XJ05	A	00350	XJCQ	A	0034B	XJMP	A	0034C
XL	A	00001	XLCONT	A	0030A	XL.10	A	00308	XL.20	A	0030D
XOR	A	0001C	XORI	A	0001B	XSCONT	A	00318	XS.10	A	00316
XS.20	A	0031A	XU	A	00000	Y	A	00003	YEXP	A	0000F
YL	A	0000A	YU	A	0000B	Z	A	00005	ZA	A	00004
ZB	A	00003	ZDIM	A	0072D	ZL	A	0000E	ZQ	A	00002
ZREG	A	00004	ZU	A	0000D	.BLE	A	002EB	.CPM	A	0027D
.CPUID	A	00303	.DAD	A	00202	.DC0	A	00264	.DC0EQ	A	00267
.DCOLT	A	00266	.DDE	A	00211	.DDI	A	00239	.DDIR	A	0023B
.DDS	A	00216	.DIN	A	0020F	.DIS	A	00213	.DMP	A	0021E
.DNG	A	0020D	.DSB	A	00205	.DSBR	A	0020A	.DSKIPZ	A	00219
.ENTC	A	00293	.ENTN	A	0028E	.ENTN05	A	00292	.ENTP	A	00297
.ENTR	A	0029A	.ENTR0	A	0029C	.ENTRL	A	0029D	.ENTRL5	A	002A0
.ENTRSUB	A	002A7	.EXIT0	A	0043B	.EXIT1	A	00443	.EXIT2	A	00445
.FAD	A	004A9	.FDV	A	004DB	.FIX	A	0047F	.FIXD	A	0048C
.FLTD	A	00493	.FLUN	A	002B3	.FMP	A	004C7	.FSB	A	004A2
.FWID	A	00304	.IMAP	A	006E8	.INTR	A	002A5	.IRES	A	006E6
.JMAP	A	006ED	.JRES	A	006EB	.LBP0	A	006CF	.LBPR	A	006C6
.LPX	A	006C8	.LPXR	A	006CB	.NGL	A	002F6	.PACK	A	002B8
.PCALI	A	00414	.PCALN	A	0042B	.PCALR	A	00423	.PCALV	A	0041A
.PCALX	A	0041B	.PMAP	A	006DE	.PWR2	A	002E0	.SDSP	A	0040D
.SETP	A	0026D	.SIP	A	00305	.TADD	A	00572	.TDIV	A	00680
.TFTD	A	00559	.TFTS	A	0052D	.TFXD	A	00537	.TFXS	A	0051B
.TMPY	A	00626	.TSUB	A	00565	.WFI	A	00306	.WFI2	A	00307
.XFER	A	003FB	..FCM	A	002AE	..TCM	A	002FF			

Vol. 2
MANUAL PART NO. 02156-90003
Printed In U.S.A. March 1983
U0684

HEWLETT-PACKARD COMPANY
Data Systems Division
11000 Wolfe Road
Cupertino, California 95014