

LPA-11

SYSTEM EXERCISER
MD-11-DRLPA-A

EP-DRLPA-A-DL
COPYRIGHT © 77-78
FICHE 1 OF 2

MAR 1978
digital
MADE IN USA

This microfiche card contains a grid of 100 frames of data, arranged in 10 rows and 10 columns. Each frame displays a different set of data, likely related to system exercises or performance metrics. The data is presented in a structured format, possibly as tables or lists, with varying amounts of text and numerical values. The frames are densely packed and cover the majority of the card's surface.

LPA-11

SYSTEM EXERCISER
MD-11-DRLPA-A

EP-DRLPA-A-DL
COPYRIGHT © 77-78
FICHE 2 OF 2

MAR 1978
digital
MADE IN USA

EOF1000000001

00010000

780223

IDENTIFICATION11

HDR1DRLPASEQ

00010000

780223
SEQ 0001

PRODUCT CODE: MAINDEC-11-DRLPA-A-D
PRODUCT NAME: LPA-11 SYSTEM EXERCISER
DATE CREATED: JAN. 1978

COPYRIGHT (C) 1977, 1978
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	EQUIPMENT
2.2	STORAGE
3.0	LOADING PROCEDURE
3.1	METHOD
3.2	NON-STANDARD ADDRESS, VECTOR
4.0	STARTING PROCEDURE
4.1	CONTROL SWITCH SETTINGS
4.2	STARTING ADDRESS
4.3	PROGRAM AND/OR OPERATOR ACTION
5.0	OPERATING PROCEDURE
5.1	SWITCH REGISTER FUNCTION
5.2	SCOPE LOOPS
5.3	PROGRAM AND/OR OPERATION ACTION
6.0	ERRORS
6.1	ERROR PRINTOUT
7.0	RESTRICTIONS
8.0	MISCELLANEOUS
8.1	POWER FAIL
8.2	XXDP, ACT, APT
8.3	EXECUTION TIME
8.4	LPA-11 (SYSTEM) DIAGNOSTIC SUMMARY
8.5	LPA-11 VERSION 4 MICRO-CODE ERROR SUMMARY
8.6	LPA-11 VERSION 4 MICRO-CODE ERROR LIST
8.7	LPA-11 (KMC-11) REGISTER DEFINITIONS
8.8	MICRO-CODE PROGRAMS
8.9	ILLEGAL INTERRUPTS ON PROGRAM START

1.0 ABSTRACT

THIS PROGRAM WAS DESIGNED TO EXERCISE THE LPA-11XX SUBSYSTEM. IT IS DIVIDED INTO TWO SECTIONS. THE FIRST SECTION EXERCISES EACH INDIVIDUAL HARDWARE COMPONENT ON THE SYSTEM. PLEASE NOTE THAT THE DEFAULT SYSTEM WHICH THE PROGRAM USES, HAS ONE AD11K AND ONE KW11K ON IT. THE USER MUST INFORM THE PROGRAM AS TO ANY DIFFERENCES IN CONFIGURATION. IF THE PROGRAM DETECTS A HARDWARE PROBLEM, IT INFORMS THE USER OF IT. THE USER SHOULD RUN THE DIAGNOSTIC DESIGNED TO DIAGNOSE THE SECTION OF HARDWARE THAT FAILED. (EXAMPLE IF DRLPA INFORMED THE USER THAT THE AD11K FAILED, THE USER SHOULD RUN THE LPA/AD11K DIAGNOSTIC.)

THE SECOND PART OF THE DIAGNOSTIC IS DESIGNED TO RUN WITH USER (KMC) MICRO-CODE. THIS IS THE FIRST (AND ONLY) TIME THAT M8254 (IPBM) INTERRUPT ARBITRATION LOGIC IS CHECKED. IF ANY PROBLEMS OCCUR HERE, ITS A GOOD BET THAT THE M8254 MODULE IS BAD. IF NOT, YOU'RE GOING TO HAVE TO CABLE THE I/O BUS TO THE UNIBUS IN ORDER TO RUN THE MORE DETAILED DIAGNOSTIC AVAILABLE FOR THE OPTIONS.

GOOD SCOPE LOOPS ARE HARD TO COME BY SINCE THIS PROGRAM MUST TRY TO KEEP THREE PROCESSORS IN SYNC.

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. PDP-11 FAMILY COMPUTER WITH 16K OF MEMORY (OR MORE) AND CONSOLE I/O FACILITIES (I.E. TTY)
2. LPA-11X TO BE EXERCISED
3. (OPTIONAL) KW11L OR KW11P (IMPROVES PROGRAM EXECUTION)

2.2 STORAGE

THIS PROGRAM OCCUPIES AND USES 16K OF MEMORY.

NOTE

IF 20K OR MORE MEMORY IS AVAILABLE, THIS PROGRAM WILL RUN A/D SAMPLING AT MAXIMUM SPEED.

3.0 LOADING PROCEDURE

3.1 METHOD

STANDARD PROCEDURE FOR NORMAL BINARY PROGRAM SHOULD BE FOLLOWED.
THIS PROGRAM IS SUPPLIED ON MULTI-MEDIA AND CAN BE LOADED BY
XXDP,ACT, OR APT.

3.2 NON-STANDARD ADDRESS,VECTOR, CONFIGURATION

THIS PROGRAM IS SET UP TO CHECK AN LPA-11X WITH STANDARD SET-UP
AS LISTED BELOW. IT IS IMPORTANT THAT IF THE EQUIPMENT ADDRESSES
VARY, THAT YOU ONLY CHANGE THESE LOCATIONS.

TAG	ADDRESS	CONTENTS	COMMENTS
---	-----	-----	-----
\$BASE:	001250	170460	::BASE ADDRESS OF EQUIPMENT
\$VECT1:	001244	000300	::VECTOR LOCATION

THE FOLLOWING ARE A LIST OF DEVICE ADDRESSES. YOU MAY CHANGE THE
ADDRESS OF A DEVICE ONLY BY MODIFYING THE FOLLOWING LIST. DO NOT
CHANGE ANY OTHER ADDRESSES!

AD11K:	1566	170400	;AD11K ADDRESS.
KW11K:	1570	170404	;KW11K ADDRESS.
DR11K1:	1572	167770	;DR11K #1 ADDR.
AA11K:	1574	170416	;AA11 ADDRESS.
AD11K2:	1576	170440	;AD11K #2 ADDRESS.
DR11K2:	1600	167760	;DR11K #2 ADDRESS.
DR11K3:	1602	167750	;DR11K #3 ADDRESS.
DR11K4:	1604	167740	;DR11K #4 ADDRESS.
DR11K5:	1606	166730	;DR11K #5 ADDRESS.
AR11:	1610	170400	;AR11 ADDRESS.
LPS11:	1612	170400	;LPS11 BASE ADDRESS.

SRI INFORMS THIS DIAGNOSTIC AS TO WHAT DEVICES ARE ON THE I/O BUS. SRI DEFAULTS TO 1 KW11K AND 1 AD11K IF YOUR CONFIGURATION IS DIFFERENT, YOU MUST CHANGE THIS LOCATION.

WORD BIT=1	OCTAL	DEVICE
0	000001	1ST AD11K
1	000002	1ST KW11K
2	000004	1ST DR11K
3	000010	1ST AA11K
4	000020	2ND AD11K#2
5	000040	2ND DR11K
6	000100	RESERVED
7	000200	3RD DR11K
8	000400	4TH DR11K
9	001000	5TH DR11K
10	002000	AR11
11	004000	RESERVED
12	010000	LPSAD (LPS A/D)
13	020000	LPSKW (LPS REAL TIME CLOCK)
14	040000	LPSVC (LPS D/A)
15	100000	LPSDR (LPS DIGITAL J/O)

SR1: 1562 003 ;DEVICE PRESENT FLAG DEFAULT
;IS 1 KW11K, 1 AD11K

SR2 INFORMS THIS DIAGNOSTIC AS TO HOW THE DEVICES SELECTED BY SRI ARE SET-UP FOR TEST. WHILE NO SPECIAL SETUP 'S REQUIRED TO RUN THIS TEST, THE DEPTH OF COVERAGE WILL INCREASE IF SPECIAL SETUPS ARE PERFORMED.

WORD BIT=1	OCTAL	FUNCTION
0	000001	AD11K HAS G5036 WRAP-AROUND MODULE.
2	000004	DR11K #1 HAS LOOP BACK CABLE
3	000010	AA11K HAS SCOPE DISPLAY (VISUAL TEST).
4	000020	AD11K #2 HAS G5036 WRAP AROUND MODULE.
5	000040	DR11K #2 HAS LOOPBACK CABLE
7	000200	DR11K #3 HAS LOOP BACK CABLE.
8	000400	DR11K #4 HAS LOOP BACK CABLE.
9	001000	DR11K #5 HAS LOOP BACK CABLE.
10	002000	AR11 HAS G5034 WRAP AROUND MODULE.
11	004000	AR11 HAS SCOPE DISPLAY (VISUAL TEST)
14	040000	LPS-11 D/A HAS SCOPE DISPLAY (VISUAL TEST)
15	100000	LPS-11 DIGITAL I/O HAS LOOP BACK CABLE

SR2: 1564 0 ;DEVICE SETUP REG.

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

BEFORE STARTING THE DIAGNOSTIC, SET ALL SWITCH REGISTER BITS AS DESIRED, SEE SECTION 5.1.

4.2 STARTING ADDRESS

200 START OF TEST

4.3 PROGRAM AND/OR OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY
2. LOAD ADDRESS 200
3. SET SWITCH REGISTER TO DESIRED SETTING
4. SET UP LOCATIONS SR1: AND SR2: TO REFLECT THE SYSTEM CONFIGURATION.
5. START PROGRAM

5.0 OPERATING PROCEDURE

5.1 SWITCH REGISTER FUNCTION

SWR BIT	OCTAL	FUNCTION WHEN SET
-----	-----	-----
15	100000	HALT ON ERROR
14	040000	LOOP ON TEST
13	020000	INHIBIT ERROR TYPEOUT
11	004000	INHIBIT TEST ITERATIONS
10	002000	RUN ONLY DEDICATED MODE USER MICRO-CODE
9	001000	RUN ONLY MULTIUSER MODE USER MICRO-CODE
8	000400	RUN TEST SELECTED BY <?:0>

5.2 SCOPE LOOPS

IF AN ERROR OCCURS AND THE USER WISHES TO SCOPE THE ERROR SWITCH REG. BIT 15 SHOULD BE SET TO HALT ON ERROR. WHEN THE CPU IS HALTED ON ERROR, SWITCH REG BIT 14 (LOOP ON TEST) AND SWR BIT 13 (INHIBIT ERROR TYPEOUT) SHOULD BE SET. SWR BIT 15 SHOULD BE CLEARED AND CPU SHOULD BE CONTINUED.

NOTE

SOME SCOPE LOOPS MAY BE IMPOSSIBLE TO OBTAIN OR NOT REPEATIBLE DUE TO THE FACT THAT THREE ASYNCHRONOUS CPUS ARE RUNNING TO GENERATE THE ERROR.

5.3 PROGRAM AND/OR OPERATOR ACTION

1. WHEN THE PROGRAM IS INITIALLY STARTED IT WILL TYPE:

MD-11-DRLPA-A

THE FIRST "PASS" THROUGH THE PROGRAM IS QUICK VERIFY OR A SHORT ONE. ALL OTHER PASSES WILL ITERATE ON EACH SUBTEST UNLESS INHIBITED.

2. THE PROGRAM PERFORMS TESTS ON THE KMC11.
3. THE PROGRAM PERFORMS TESTS ON THE MB200-YC AND MB254.
4. THE PROGRAM TESTS EACH OPTION ON THE I/O BUSS SELECTED BY THE OPERATOR.
5. THE PROGRAM LOADS USER MICRO-CODE INTO THE KMCII AND EXERCISES TOTAL LPAII-KX SYSTEM.
6. PROGRAM END PASS.

NOTE

ON ALL EVEN PASSES THROUGH THE PROGRAM, THE PROGRAM SELECTS MULTIUSER MICRO-CODE FOR USER MICRO CODE; AND ON ALL ODD NUMBERED PASSES THE PROGRAM SELECTS DEDICATED MODE MICRO-CODE FOR USER MICRO-CODE. THE OPERATOR MAY INFORM THE PROGRAM TO ONLY RUN ON VERSION OF USER-MICRO-CODE BY USE OF THE SWITCH REGISTER (SEE SECTION 5.1)

6.0 ERRORS

6.1 ERROR PRINT-OUT

PRINTOUT VARIES WITH THE ERROR DETECTED. THE ERROR PC TYPED OUT IS THE ACTUAL LOCATION OF THE ERROR CALL.

THE FOLLOWING IS A LIST OF LABELS ASSOCIATED WITH LPA11 FUNCTIONAL ERROR.

TSTWD - TEST IN WHICH THE ERROR WAS DETECTED.

USJNO - USER JOB NUMBER ASSIGNED TO THE JOB BY THE MICRO-CODE.

ALPCO - LPA11 CONTROL OUT REGISTER CONTENTS.

ALPCI - LPA11 CONTROL IN REGISTER CONTENTS.

ALPSO - LPA11 STATUS OUT REGISTER CONTENTS.

NOTE

ERROR REPORTS MAY OR MAY NOT SPECIFY A DEVICE. IF SO, THAT DEVICE IS NOT NECESSARILY THE FAULTY UNIT. HOWEVER, IT IS THE DEVICE UNDER TEST WHEN THE ERROR WAS DETECTED. FOR FURTHER INFORMATION, THE OPERATOR MUST CONSULT THE LISTING.

7.0 RESTRICTIONS

NONE.

8.0 MISCELLANEOUS

8.1 POWER FAIL

THIS PROGRAM WILL NOT SUPPORT POWER FAILURES. IF A POWER FAILURE OCCURS, YOU MUST RELOAD AND RESTART THIS PROGRAM.

8.2 XXDP,ACT,APT

THIS PROGRAM IS CHAINABLE UNDER XXDP, ACT, OR APT. ALTHOUGH "APT HOOKS" HAVE BEEN INSTALLED, THEY HAVE NOT BEEN TESTED.

8.3 EXECUTION TIME

THE EXECUTION TIME WILL VARY BETWEEN CPUS. EXECUTION TIME ALSO VARIES WITH THE NUMBER AND TYPE OF OPTIONS ON THE LPA-11XX SYSTEM. THE APPROXIMATE TIMES ARE LISTED BELOW:

1.0 MINUTE (60 SEC) -NO ERRORS-ITERATIONS INHIBITED
3.0 MINUTE (180 SEC) -NO ERRORS-WITH ITERATIONS.
(LISTED UNDER MIS.)

8.4 LPA11 (SYSTEM) DIAGNOSTIC SUMMARY

DIAGNOSTICS FOR THE LPA11 ARE WRITTEN AT THREE LEVELS: (1) TOTAL PDP-11 SYSTEM, (2) LPA11 SYSTEM; AND, (3) LPA11 OPTIONS.

LEVEL 1 IS DESIGNED TO ISOLATE A FAILURE TO THE LPA11 SYSTEM. ALL OPTIONS ON THE PDP-11 ARE EXERCISED.

LEVEL 2 DIAGNOSTICS ISOLATE A FAILURE TO THE INDIVIDUAL OPTION WITHIN THE LPA11. THE LEVEL 2 DIAGNOSTIC IS MD-11-DRLPA. WHEN THE USER RUNS DRLPA HE CAN GENERALLY TELL WHICH OPTION DIAGNOSTIC (LEVEL 3) TO RUN NEXT. MB254 AND MB200-YC ERRORS MAY "LOOK" ALIKE AND DRLPA MAY NOT BE ABLE TO DISTINGUISH BETWEEN THEM. ARBITRATION ERRORS WILL NOT BE DETECTED BY THIS DIAGNOSTIC.

LEVEL THREE DIAGNOSTICS AID IN DETERMINING IF THE ERROR WAS IN FACT ON THE OPTION THE DRLPA SPECIFIED. THE USER MAY "LOOP" ON THE ERROR. WITHIN LEVEL THREE, THERE ARE TWO GROUPS OF DIAGNOSTICS. THE FIRST GROUP REQUIRES NO "EXTRA" WORK BY THE USER IN ORDER TO RUN. GROUP "A" DIAGNOSTICS DO NOT CHECK ARBITRATION, AND REQUIRE EXTRA TIME FOR EXECUTION. THE SECOND GROUP (GROUP "B") REQUIRES THAT THE USER RECONFIGURE THE PDP-11 SYSTEM. THIS RECONFIGURATION INVOLVES CABLING THE UNIBUS TO THE LPA'S I/O BUS.

THE DIAGNOSTIC FOR THE MB254 FALLS INTO THE GROUP "B" CATEGORY.

8.5 LPA11 VERSION 3 MICRO-CODE ERROR SUMMARY

MICRO-CODE ERRORS ARE DEFINED AS ANY ERROR RETURNED TO THE USER BY THE MICRO-CODE THROUGH THE LPA11 STATUS REGISTER.

WITHIN THE DIAGNOSTIC WE CHECK TO BE SURE THAT WE CAN GENERATE SOME OF THESE ERRORS. OTHERS MAY COME UP AT ANY TIME WHILE THE LPA11-XX MICRO-CODE IS BEING RUN.

8.5.1 FATAL HARDWARE ERRORS

THE FATAL HARDWARE ERRORS CONSIST OF DATA ERRORS, IMPROPER STATUS CONDITIONS, OR MALFUNCTIONS WHICH ARE DETECTED DURING THE INITIALIZATION OF THE LPA11 SUBSYSTEM OR DURING THE PROCESSING OF DATA. NO SUBSEQUENT DATA WILL BE TRANSFERRED AND THE LPA11 WILL NOT RESPOND TO ADDITIONAL COMMANDS FROM THE PDP-11. THE LPA11 MUST BE RE-INITIALIZED PRIOR TO ACCEPTING ANOTHER COMMAND FROM THE PDP-11.

THESE ERRORS MAY COME UP IF THERE IS AN ARBITRATION ERROR.

8.5.2 START REQUEST ERRORS

THE START REQUEST ERRORS CONSIST OF ERRORS DETECTED BY THE MASTER DURING THE PROCESSING OF A NEW PDP-11 COMMAND TO THE LPA11.

8.5.3 USER REQUEST ERRORS

THE USER REQUEST ERRORS ARE SPECIFIED BY ERROR CODE 0-4 AND ERROR STATUS BITS 0 AND 1 OF THE STATUS OUT BYTE 3. THE REQUEST IS DEALLOCATED AND MUST BE REISSUED IN ORDER TO RESUME. THE USER REQUESTS ERROR CODES FROM 240 TO 247 ARE GENERATED BY THE MASTER MICROPROCESSOR AND ERROR CODES FROM 250 TO 254 AND 257 ARE GENERATED BY CONDITIONS DETECTED IN THE SLAVE MICROPROCESSOR AND TRANSFERRED TO THE CSR OF THE MASTER. IN THE MULTI REQUEST MODE THE USER FOR WHICH THE ERROR APPLIES IS IDENTIFIED BY THE USER INDEX CODE OF THE CONTROL OUT WORD BYTE 2.

8.5.4 NORMAL STATUS RETURNS

THE NORMAL STATUS RETURNS ARE INDICATED BY ERROR CODE 0-4. THE STATUS RETURNS INDICATE A FULL OR OVERRUN CONDITION OF THE PDP-11 MEMORY BUFFERS ASSIGNED TO STORE DATA FROM THE LPA11. THE BUFFER ADDRESSES ARE SPECIFIED BY THE RDA INFORMATION CONTAINED IN THE I/O DEVICE START COMMAND. THE BUFFER FULL AND OVERRUN STATUS CAN BE SPECIFIED AS A FATAL OR NON FATAL CONDITION BY THE RDA VALID BUFFER MASK CONFIGURATION OF THE USER STATUS WORD. ANY OF THE THREE NORMAL STATUS RETURNS WILL CAUSE AN LPA11 CONTROL OUT INTERRUPT REQUEST TO BE GENERATED. THE BUFFER FULL CONDITION EXISTS WHEN THE LPA11 HAS FILLED A ASSIGNED BUFFER WITH DATA. THE BUFFER OVERRUN CONDITION IS A RESULT OF ALL THE ASSIGNED BUFFERS BEING LOADED BEFORE THE PDP-11 HAS PROCESSED OR TRANSFERRED THE DATA FROM THE BUFFER. IF THE BUFFER OVERRUN CONDITIONS OCCUR CONTINUALLY; A USER REQUEST ERROR WILL RESULT.

THE LPA11-KX DIAGNOSTIC KIT WILL INCLUDE:

OPTION -----	GROUP -----	DIAG. # -----	DIAG. TITLE -----
LPA11-KX	LEVEL 2 DIAG.	MD-11-DRLPA	LPA11-KX SYSTEM
M8254	"B"	MD-11-DRLPN	M8254 (IPBM) DIAG.
AA11-K	A	MD-11-DRLPB	AA11-K DIAG.
	B	MD-11-DZAAC	AA11-K DIAG.
AR11-K	A	MD-11-DRLPC	LPA/AR11 DIAG. #1
	A	MD-11-DRLPD	LPA/AR11 DIAG. #2
	A	MD-11-DRLPE	LPA/AR11 DIAG. #3
	B	MD-11-DZARA	AR11 DIAG. #1
	B	MD-11-DZARB	AR11 DIAG. #2
	B	MD-11-DZARC	AR11 DIAG. #3
	DR11-K	A	MD-11-DRLPF
B		MD-11-DZDRG	DR11-K DIAG.
KW11-K	A	MD-11-DRLPG	LPA/KW11-K DIAG.
	B	MD-11-DZKWK	KW11-K DIAG.
LPS11	A	MD-11-DRLPH	LPA/LPS11 DIAG. #1
	A	MD-11-DRLPI	LPA/LPS11 DIAG. #2
	A	MD-11-DRLPJ	LPA/LPS11 DIAG. #3
	B	MD-11-DZLPC	LPS11 DIAG. #1
	B	MD-11-DZLPD	LPS11 DIAG. #2
	B	MD-11-DZLPI	LPS11 DIAG. #3
AD11-K	A	MD-11-DRLPK	LPA/AD11-K DIAG.
	B	MD-11-DZADL	AD11-K DIAG.
M8200-YC	B	MD-11-DRLPL	LPA/M8200-YC BASIC MICRO-CPU R/W TEST
	B	MD-11-DRLPM	LPA/M8200-YC JMP+ROM READ TEST

8.6 LPA-11 VERSION 3 MICRO-CODE ERROR LIST

***** FATAL HARDWARE ERRORS *****

322 --- ADDRESS OF NON-EXISTENT DEVICE
 340 --- SLAVE POWER LOW
 341 --- MASTER FIFO READ / WRITE ERROR
 342 --- I/O BUS SACK TIME OUT
 343 --- MASTER INITIAL CONDITION ERROR
 BYTE 6 --- EXPECTED STATUS
 BYTE 7 --- BAD STATUS
 344 --- MASTER / SLAVE VERSION ERROR
 BYTE 6 --- MASTER VERSION NUMBER
 BYTE 7 --- SLAVE VERSION NUMBER
 345 --- SLAVE COLD START TIME-OUT
 346 --- FATAL SLAVE ERROR
 BYTE 6 --- 200 --- SLAVE FIFO READ / WRITE ERROR
 201 --- KW11-K CLOCK "A" OVERRUN
 202 --- SLAVE FIRMWARE FIFO SEQUENCE ERROR
 347 --- FPATH DATA ERROR
 BYTE 6 --- BAD VALUE
 BYTE 7 --- EXPECTED VALUE
 350 --- FIFO DATA ERROR
 BYTE 6 --- BAD VALUE
 BYTE 7 --- EXPECTED VALUE

***** START REQUEST ERRORS *****

300 --- NO ROOM FOR REQUEST
 301 --- "GO" SET WITH "RDY IN" CLEAR
 302 --- MULTI-USER REQUEST WITH DEDICATED MODE MICRO CODE LOADED
 304 --- DEDICATED MODE REQUEST WITH MULTI-USER MICRO CODE LOADED
 306 --- "START COMMAND WITH NO "INITIALIZE"
 310 --- MULTIPLE "INITIALIZE" COMMANDS
 312 --- "STOP" COMMAND WITH USER NOT ACTIVE
 314 --- ODD ADDRESS SPECIFIED FOR REQUEST DESCRIPTOR ARRAY
 316 --- "INITIALIZE" WITH WRONG VERSION SPECIFIED
 320 --- "START" FOR DEVICE NOT IN CONFIGURATION
 322 --- "START" WITH ILLEGAL FUNCTION SPECIFIED
 324 --- NON EXISTANT MEMORY IN REQUEST DESCRIPTOR ARRAY
 325 --- ODD ADDRESS SPECIFIED FOR BUFFER OR USER STATUS WORD
 326 --- DEVICE NOT FOUND ON I/O BUS DURING "INITIALIZE" COMMAND
 BYTE 6,7 --- ADDRESS OF NON EXISTANT DEVICE

***** USER REQUEST ERRORS *****

240 --- NON FATAL ERROR COUNT EXCEEDED
241 --- ERROR STATUS OVERRUN
242 --- NON EXISTANT MEMORY IN RANDOM CHANNEL LIST
243 --- BUFFER OVERRUN / UNDERRUN
244 --- NON EXISTANT MEMORY IN BUFFER
245 --- NON EXISTANT MEMORY IN USER STATUS WORD
246 --- INVALID BUFFER INDEX SPECIFIED IN USER STATUS WORD
247 --- MASTER FIFO 7/8 FULL
250 --- REQUEST TERMINATED BY USER STATUS WORD REQUEST
251 --- NO FIRMWARE FIFO BUFFER AVAILABLE FOR REQUEST
252 --- SLAVE FIFO 7/8 FULL
253 --- RANDOM CHANNEL ADDRESS UNDERRUN
254 --- DATA UNDERRUN
255 --- NON EXISTANT CHANNEL OR DEVICE
260 --- MULTIPLE EXTERNAL TRIGGER DIGITAL OUTPUT REQUESTS

***** NORMAL STATUS RETURNS *****

000 --- START REQUEST PROCESSED
001 --- BUFFER FULL
002 --- BUFFER OVERRUN / UNDERRUN

8.7 LPA-11 (KMC-11) REGISTER DEFINITIONS

THE CONTROL AND STATUS INFORMATION, TRANSFERRED BETWEEN THE PDP-11 UNIBUS AND THE LPA-11 SUBSYSTEM, IS STORED IN THE CONTROL AND STATUS REGISTERS (CSR) OF THE MASTER MICROPROCESSOR (M8204). THE CSR'S CONSIST OF EIGHT, 8-BIT BYTES IMPLEMENTED AS FOUR 16-BIT WORDS OF MULTIPORT RANDOM ACCESS MEMORY (RAM'S). THE CSR LOCATIONS ARE ADDRESSABLE FROM EITHER THE PDP-11 OR THE M8204 MICROPROCESSOR PROGRAM. THE UNIBUS ADDRESSES FOR EACH CSR IS AS FOLLOWS:

MASTER MICROPROCESSOR CSR

BYTE	UNIBUS	NUM	WORD
0 (LOW)	76 XXX 0	LPCI	1
1 (HIGH)	76 XXX 1	MAIN	1
2 (LOW)	76 XXX 2	LPCO	2
3 (HIGH)	76 XXX 3	LPSO	2
4 (LOW)	76 XXX 4	LPADL	3
5 (HIGH)	76 XXX 5		4
6 (LOW)	76 XXX 6	LPMS1	4
7 (HIGH)	76 XXX 7		4

CONTROL IN (BYTE 0)

THE CONTROL IN BYTE CONSISTS OF 8 BITS AT ADDRESS 76XXX0 USED BY THE PDP-11 PROGRAM TO INITIATE THE TRANSFER OF COMMANDS TO THE LPA-11 SUBSYSTEM. THE FORMAT AND DESCRIPTION OF BYTE INFORMATION IS SHOWN ON FIGURE 8-1.

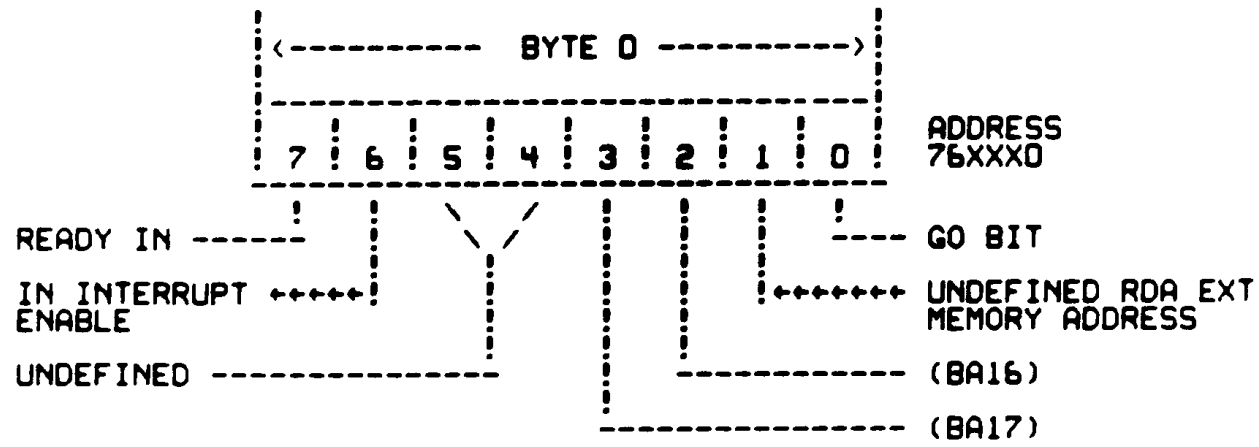


FIGURE 8-1
CONTROL IN (BYTE 0)

BIT ---	DESCRIPTION -----
0	<p>GO BIT- INITIAL CONDITION IS LOW (0). SET TO HIGH (1) BY THE PDP-11 PROGRAM TO INDICATE THAT A VALID REQUEST DESCRIPTOR ARRAY (RDA) ADDRESS IS AVAILABLE FOR PROCESSING BY THE LPA-11. CLEARED LOW (0) BY THE MASTER MICROPROGRAM AFTER THE USER'S REQUEST IS INITIATED.</p> <p style="padding-left: 40px;">READ - LPA-11 MICROPROGRAM WRITE - PDP-11 PROGRAM</p>
1	NOT SPECIFIED
2,3	<p>BUS ADDRESS (BA16,BA17) - EXTENDED MEMORY BITS FOR REQUEST DESCRIPTOR ARRAY (RDA) ADDRESS (BYTES 4 AND 5) OF CSR.</p> <p style="padding-left: 40px;">READ - PDP-11 PROGRAM WRITE - PDP-11 PROGRAM</p>
4,5	NOT SPECIFIED (MST BE LOW)
6	<p>IN INTERRUPT ENABLE - INITIAL CONDITION IS LOW (0). SET TO HIGH (1) BY PDP-11 PROGRAM TO ENABLE INTERRUPT REQUEST TO BE GENERATED WHEN THE READY IN, BIT 07, IS HIGH (1) OR DURING A LOW TO HIGH TRANSITION.</p> <p style="padding-left: 40px;">READ - PDP-11 PROGRAM WRITE - PDP-11 PROGRAM</p> <p>AN INTERRUPT OCCURS AT VECTOR ADDRESS +4. A READY IN INTERRUPT MAY OCCUR EITHER IMMEDIATELY OR AFTER A USEC DELAY. THE LPA-11 WILL CAUSE AN OUT INTERRUPT OR OUT STATUS BEFORE THE READY IN BIT IS SET IF BOTH ARE PENDING SIMULTANEOUSLY.</p>
7	<p>READY IN - INITIAL CONDITION IS LOW 0. SET TO HIGH (1) BY MICROPROCESSOR WHEN READY TO ACCEPT USER'S REQUEST. CLEARED LOW 0 AFTER REQUEST IS GRANTED.</p> <p style="padding-left: 40px;">READ - PDP-11 PROGRAM WRITE - MICROPROCESSOR</p>

MAINTENANCE (BYTE 1)

THE MAINTENANCE BYTE IS AN EIGHT BIT, HIGH BYTE AT ADDRESS 76XXX1 WHICH CAN BE MONITORED BY THE PDP-11 PROGRAM DURING DIAGNOSTIC FUNCTIONS. REFER TO THE PROGRAMMING SECTION 4 OF THE KMC-11 MAINTENANCE MANUAL FOR A DESCRIPTION OF THE BITS NOT DEFINED. FIGURE 8-2 SHOWS THE BIT CONFIGURATION ON THE MAINTENANCE BYTE.

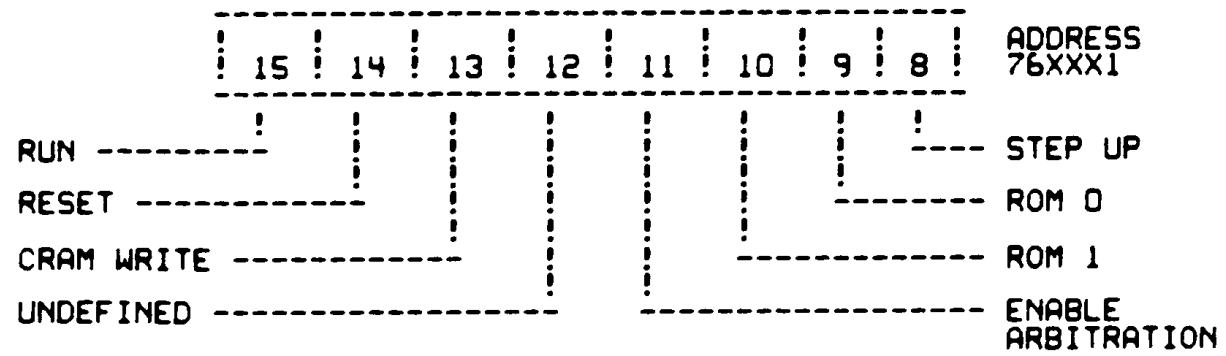


FIGURE 8-2
MAINTENANCE (BYTE 1)

<u>BIT</u>	<u>DESCRIPTION</u>
8,9,10	REFER TO KMC-11 MAINTENANCE MANUAL
11	ENABLE ARBITRATION - SET HIGH (1) BY PDP-11 PROGRAM TO ALLOW THE 78254 MODULE TO ARBITRATE NON-PROCESSOR REQUESTS (NPR) AND BUS REQUESTS BETWEEN THE SLAVE PROCESSOR AND DEVICES ON THE I/O BUS. WHEN CLEARED LOW (0) BY PDP-11 PROGRAM THE ARBITRATION LOGIC IS INHIBITED. READ - PDP-11 PROGRAM AND MICROPROCESSOR WRITE - PDP-11 PROGRAM
12,13	REFER TO KMC-11 MAINTENANCE MANUAL
14	RESET - SET HIGH (1) BY PDP-11 PROGRAM TO CLEAR ALL PERTINENT REGISTERS IN THE LPA-11 SUBSYSTEM. READ - LPA-11 MICROPROGRAM WRITE - PDP-11 PROGRAM ONLY
15	REFER TO KMC-11 MAINTENANCE MANUAL

CONTROL OUT (BYTE 2)

THE CONTROL OUT BYTE CONSISTS OF 8 BITS AT ADDRESS 76XXX2 AND IS USED BY THE LPA-11 TO INDICATE TO THE PDP-11 THE AVAILABILITY OF THE LPA STATUS INFORMATION. THE FOMAT AND DESCRIPTION OF THE BYTE INFOMATION IS SHOWN ON FIGURE 8-3.

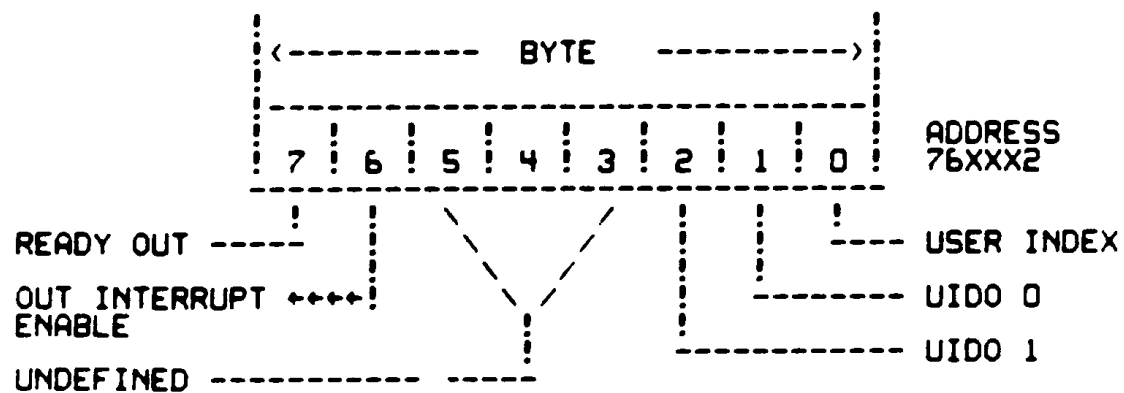


FIGURE 8-3
CONTROL OUT (BYTE 2)

BIT ---	DESCRIPTION -----
0,1,2	USER'S INDEX - AN OCTAL CODE (0-7) ASSIGNED BY THE MASTER MICROPROCESSOR IN THE MULTI REQUEST MODE TO IDENTIFY THE STATUS AS RELATED TO UP TO EIGHT USER REQUESTS.
3,4,5	NOT SPECIFIED
6	OUT INTERRUPT ENABLE - SET HIGH (1) BY MASTER MICROPROCESSOR PROGRAM TO ALLOW AN INTERRUPT REQUEST TO BE GENERATED AT VECTOR +00 WHEN THE READY OUT (BIT 07) IS SET. WRITE - PDP-11 PROGRAM
7	READY OUT - SET HIGH (1) BY THE MASTER MICROPROCESSOR PROGRAM TO INDICATE THAT THE LPA-11 SUBSYSTEM HAS STATUS INFORMATION FOR THE PDP-11. CLEARED LOW (0) BY THE PDP-11 PROGRAM TO ACKNOWLEDGE THE RECEPTION OF THE LPA-11 STATUS INFORMATION. READ - PDP-11 PROGRAM WRITE - PDP-11 PROGRAM

USER INDEX CODE

A USER INDEX CODE IS SPECIFIED BY THE LPA-11 TO IDENTIFY THE ERROR OR STATUS INFORMATION AS RELATED TO ONE OF EIGHT USER'S REQUESTS. THE USER'S INDEX IS AN OCTAL CODE FORM 0 - 7 AND IS ASSIGNED TO THE USER IN ORDER THAT THE REQUESTS ARE GRANTED. THIS EFFECTIVELY PROVIDES A VARIABLE INDEX TO IDENTIFY FURTHER COMMUNICATIONS WITH AN ASSOCIATED USER. THE USER'S INDEX CODE IS NOT VALID FOR FATAL HARDWARE ERROR CONDITION IN LPA-11 WHICH EFFECT ALL ACTIVE USER'S AND FOR START REQUEST ERRORS WHICH OCCUR PRIOR TO ESTABLISHING THE NEW USER'S REQUEST.

STATUS OUT (BYTE 3)

THE STATUS OUT IS AN 8-BIT, HIGH BYTE AT ADDRESS 76XXX3 WHICH INDICATES STATUS AND ERROR CONDITIONS TO THE PDP-11 PROGRAM DURING THE ESTABLISHMENT OF USER REQUESTS OR DURING THE TRANSFER OF DATA. THE ERRORS ARE CLASSIFIED AS FATAL HARDWARE ERRORS WHICH RESULT IN A HALT OF THE MICROPROCESSOR OR PROGRAM AND NON FATAL ERRORS WHICH INDICATE SPECIFIC STATUS CONDITIONS WHICH HAVE OCCURRED. THE HALT CONDITION PREVENTS THE LPA-11 FROM ACCEPTING ANY ADDITIONAL USER REQUESTS OR FROM TRANSFERRING ANY DATA TO OR FROM THE PDP-11. THE STATUS OUT BYTE CONTAINS FIVE ERROR CODE BITS 8 - 12, TWO ERROR STATUS BITS 13,14 AND FATAL ERROR INDICATOR, BIT 15, AS SHOWN ON FIGURE 8-4. THE ERROR CODES LISTED ON TABLE 8-1, 8-2 AND 8-3 ARE GROUPED AS SHOWN ON THE STATUS OUT BIT CONFIGURATION.

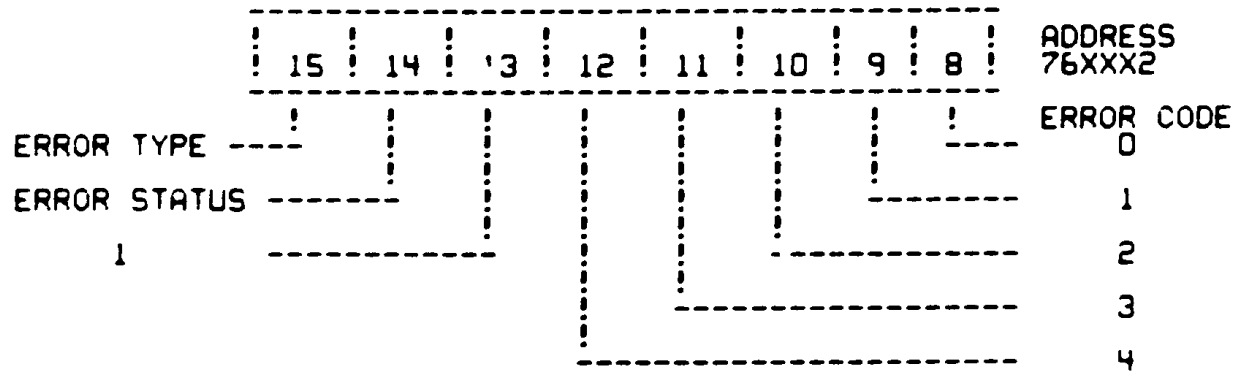


FIGURE 8-4
STATUS OUT (BYTE 3)

BIT	DESCRIPTION
---	-----
8-12	ERROR CODE 0-4 - A SUB CODE AS LISTED ON TABLE WHICH FURTHER DEFINES THE ERROR CODE INDICATED BY THE ERROR STATUS 0 (BIT 14) AND ERROR STATUS 1 (BIT 1). READ - PDP-11 PROGRAM WRITE - MASTER MICROPROCESSOR
13,14	ERROR STATUS 0 AND 1 - INDICATES THE CONDITIONS FROM WHICH THE ERROR OR STATUS ORIGINATED READ - PDP-11 PROGRAM WRITE - MASTER MICROPROCESSOR
15	STATUS/ERROR INDICATOR - A SINGLE BIT WHICH INDICATES STATUS OR FATAL ERROR CONDITIONS REQUIRING ATTENTION. A HIGH (1) INDICATES AN ERROR AND A LOW (0) INDICATES STATUS. READ - PDP-11 PROGRAM WRITE - MASTER MICROPROCESSOR

ERROR/STATUS CODES

THE ERROR/STATUS CODES INDICATED BY THE STATUS OUT BYTE 3 ARE DEFINED AS FATAL HARDWARE ERRORS, START REQUEST ERRORS, USER REQUEST ERRORS AND NORMAL STATUS RETURNS.

THE ERROR/STATUS CODES AS LISTED ON THE FOLLOWING TABLES INCLUDE ALL EIGHT BITS STARTING AT BIT 0 TO BIT 7.

REQUEST DESCRIPTOR ARRAY ADDRESS (BYTE 4 AND 5)

THE BUS ADDRESSES OF THE REQUEST DESCRIPTOR ARRAYS (RDA) IN THE PDP-11 MEMORY ARE CONTAINED WITHIN BYTES 4 AND 5 OF THE LPA-11 CSR. BYTE 4 AND 5 EACH CONTAIN 8-ADDRESS BITS (BA00-BA07 AND BA08-BA15) AS SHOWN ON FIGURE 8-5 AND BYTE 0 CONTAINS TWO ADDRESS BITS (BA16 AND BA17).

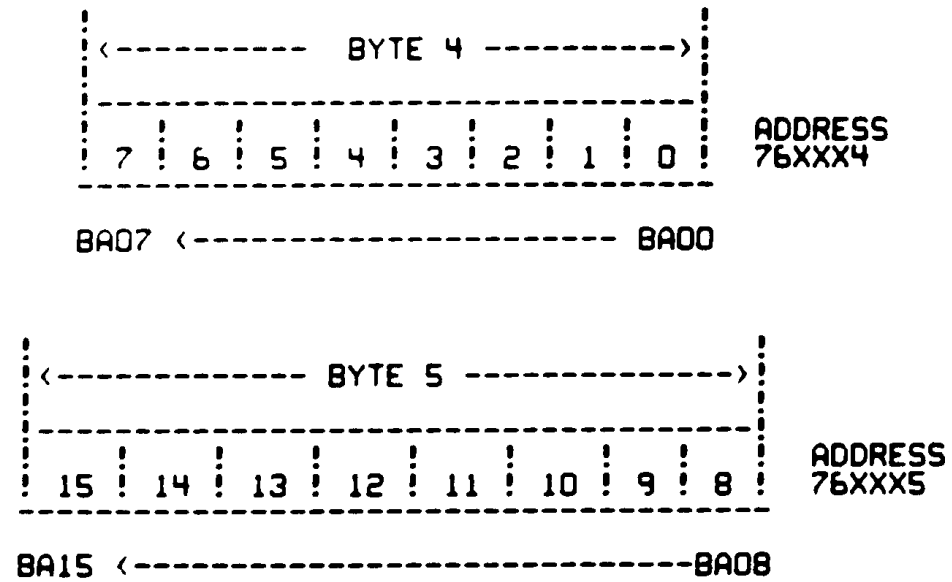


FIGURE 8-5
REQUEST DESCRIPTOR ARRAY ADDRESS

<u>BIT</u>	<u>DESCRIPTION</u>
0-7	BA00-BA07 - LOW BYTE OF THE RDA ADDRESS IN PDP-11 MEMORY
8-15	BA08-BA15 - HIGH BYTE OF THE RDA ADDRESS IN PDP-11 MEMORY.

MAINTENANCE STATUS (BYTE 6 AND 7)

THE MAINTENANCE STATUS, BYTES 6 AND 7, ARE LISTED ON TABLE 8-6 AND INDICATE THE SPECIFIC STATUS AND ERROR CHECK VALUES ASSOCIATED WITH THE FATAL HARDWARE ERRORS DEFINED ON TABLE 8-2. THE FORMAT OF THE LOW AND HIGH 8-BIT ERROR STATUS CODES ARE SHOWN ON FIGURE 8-6.

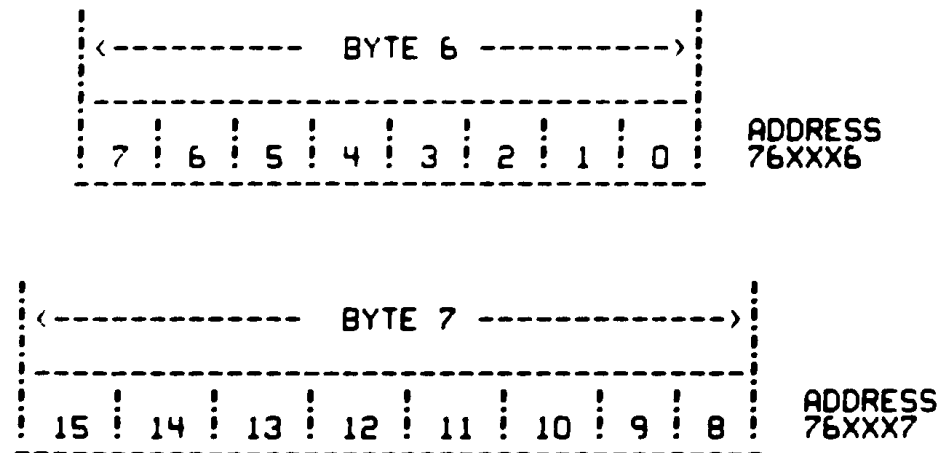


FIGURE 8-6
ERROR STATUS (BYTES 6 AND 7)

8.8 MICRO-CODE PROGRAMS

THERE ARE FOUR SEPARATE MICRO-CODE PROGRAMS LOADED INTO THE KMC11 DURING THE COURSE OF THIS DIAGNOSTIC. A BRIEF DESCRIPTION OF EACH FOLLOWS.

8.8.1 DRLPX1

THIS IS THE FIRST MICRO-CODE PROGRAM LOADED INTO THE KMC11. ITS FUNCTION IS TO VERIFY PROPER OPERATION OF THE KMC11 BOARD. IT WILL PERFORM CROM READ/WRITE TESTS, BRANCH TESTS, ALU TESTS, NPR TESTS, AND INTERRUPT TESTS. IF THE KMC11 IS FOUND TO BE FAULTY, THE OPERATOR MUST ABORT DRLPA AND RUN THE KMC11 DIAGNOSTICS.

8.8.2 DRLPX0

THE MICRO-CODE PROGRAM "DRLPX0" IS LOADED INTO THE KMC11 AT THE ONSET OF TEST #3. DRLPX0 PROVIDES COMMUNICATIONS FACILITIES BETWEEN THE KMC11 AND THE SLAVE MICRO-PROCESSOR. IT IS USED DURING TESTING OF THE SLAVE MICRO-PROCESSOR, THE IPBM, AND THE OPTIONS ON THE I/O BUS.

8.8.3 USER MICRO-CODE

THERE ARE TWO SEPARATE USER MICRO-CODE PROGRAMS LOADED INTO THE KMC11: DEDICATED AND MULTI-USER. THEIR FUNCTION IN RELATION TO THIS DIAGNOSTIC IS TO EXERCISE THE OPTIONS ON THE I/O BUS. A BRIEF DESCRIPTION OF EACH FOLLOWS.

DEDICATED MICRO-CODE

THIS PROGRAM ALLOWS HIGH SPEED SAMPLING OF A/D DEVICES ON THE I/O BUS. ONLY ONE JOB CAN BE RUN AT ANY GIVEN TIME.

MULTI-USER MICRO-CODE

WHEN RUNNING MULTI-USER MICRO-CODE, ANY NUMBER OF JOBS FROM ONE TO EIGHT MAY BE RUN AT THE SAME TIME BY THE LPA11 SUBSYSTEM. THE JOBS MAY BE DIGITAL INPUT, A/D SAMPLING, AND/OR D/A CONVERSIONS.

9.9 ILLEGAL INTERRUPTS ON START

SINCE THE KMC11 CAN NOT BE INITIALIZED UNTIL TEST #1 IS IN PROGRESS, THERE EXISTS THE POSSIBILITY THAT PREVIOUSLY LOADED MICRO-CODE WILL CAUSE A PROCESSOR INTERRUPT. THIS WILL BE INDICATED BY A PROCESSOR HALT AT THE KMC11 INTR. VECTOR LOCATION.

IN ORDER TO CORRECT THIS PROBLEM, THE OPERATOR MUST MANUALLY CLEAR ALL FOUR KMC11 REGISTERS SEEN BY THE UNIBUS. THIS IS DONE BY LOADING THE BASE ADDRESS OF THE KMC11 AND DEPOSITING ZERO'S IN FOUR CONSECUTIVE WORD LOCATIONS.

THE PROGRAM CAN THEN BE RESTARTED.

ORLPA.P11

TABLE OF CONTENTS

45	OPERATIONAL SWITCH SETTINGS
57	TRAP CATCHER
77	BASIC DEFINITIONS
190	ACT11 HOOKS
202	APT PARAMETER BLOCK
224	COMMON TAGS
268	APT MAILBOX-ETABLE
317	ERROR POINTER TABLE
497	*****
498	SR1 DEVICE PRESENT REG
499	*****
533	*****
534	SR2 DEVICE SET-UP REG
535	*****
564	*****
565	LPA-11 I/O BUS DEVICE ADDRESS DEFINITION AND LIST
566	*****
590	AD11-K ADDRESSES
600	AR11-K ADDRESSES
609	DR11K #1 ADDRESS
614	DR11K #2 ADDRESS
619	DR11K #3 ADDRESS
624	DR11K #4 ADDRESS
629	DR11K #5 ADDRESS
636	KW11K ADDRESS
650	AR11 ADDRESSES
663	LPS-11 ADDRESSES
707	INITIALIZE THE COMMON TAGS
893	TYPE PROGRAM NAME
898	GET VALUE FOR SOFTWARE SWITCH REGISTER
917	T1 *TEST THE ADDRESSABILITY OF THE KMC-11
957	T2 *TEST KMC-11 FUNCTIONS
972	KMC-11 MICRO-INITIALIZATION.
983	KMC-DIAG. SUB-TEST #1 CRAM WRITE ONES TEST
1034	KMC-DIAG SUB-TEST #2 CRAM WRITE ZEROES TEST
1083	KMC-DIAG SUB-TEST #3 BRANCH TEST
1149	KMC-DIAG SUB-TEST #4 ALU TEST
1196	KMC-DIAG SUB-TEST #5 NPR TEST ZERO TRANSFER
1267	KMC-DIAG SUB-TEST #6 NPR TEST ONES TRANSFER.
1339	KMC-DIAG SUB-TEST #7 INTERRUPT TEST.
1417	T3 *TEST SLAVE MICRO PROCESSOR START
1499	**PHASE 2 IPBM/DMC TESTING.
1500	T4 *TEST DATA LOOP BACK THROUGH IPBM,M8254 MODULE
1709	PHASE 3 I/O BUS TESTING.
1713	T5 I/O DEVICE TEST-AD11-K OPTION
1876	T6 I/O DEVICE TEST-AD11-K OPTION
2039	T7 I/O DEVICE TEST-KW11K OPTION
2237	T10 *TEST THE DR11K OPTION, #1, IF "SR1" BIT 2=1 (SET)
2384	T11 *TEST THE DR11K OPTION, #2, IF "SR1" BIT 5=1 (SET)
2531	T12 *TEST THE DR11K OPTION, #3, IF "SR1" BIT 7=1 (SET)
2678	T13 *TEST THE DR11K OPTION, #4, IF "SR1" BIT 8=1 (SET)
2825	T14 *TEST THE DR11K OPTION, #5, IF "SR1" BIT 9=1 (SET)
2973	T15 *TEST THE AA-11 OPTION, IF "SR1" BIT 3=1 (SET)
3190	T16 I/O DEVICE TEST-AR11 OPTION

3285	T17	I/O DEVICE TEST-AR11 CLOCK OPTION
3407	T20	*TEST THE AR11 DISPLAY OPTION, IF "SR1" BIT 10=1 (SET)
3555	T21	*TEST THAT THE AR11 CAN DISPLAY A SQUARE, SELECTED BY BIT 10 OF SR1,SR2
3606	T22	I/O DEVICE TEST-LPS-11 A/D OPTION
3733	T23	I/O DEVICE TEST-LPS-CLOCK OPTION
3858	T24	*TEST THE LPS I/O, IF "SR1" BIT 15=1 (SET)
4004	T25	*TEST THE LPS D/A OPTION, IF "SR1" BIT 14=1 (SET)
4158	T26	*TEST THAT THE LPS D/A CAN DISPLAY A SQUARE
4212	T27	*TEST THAT THE AR11K # CAN DISPLAY A SQUARE, SELECTED BIT 3 IN SR1,SR2
4265	T30	*TEST THAT USER MICRO-CODE CAN BE STARTED
4324	T31	*TEST FOR ERROR CODE 306
4372	T32	*TEST THAT WE CAN GENERATE ERROR CODE 314
4408	T33	*TEST THAT THE DEVICE LIST CAN BE VERIFIED
4553	T34	*TRY TO GENERATE ERROR CODE 312
4588	T35	*TRY TO GENERATE ERROR CODE 310
4647	T36	*TEST INTERRUPTS
4703	T37	*TEST THAT THE LPA-SYSTEM CLOCK CAN BE STARTED
4759	T40	*TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DR11K #1
4873	T41	*TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DR11K #2
4987	T42	*TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DR11K #3
5101	T43	*TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DR11K #4
5215	T44	*TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DR11K #5
5330	T45	*TEST THE LPA SYSTEM USER MICRO-CODE'S ABILITY TO TAKE SAMPLE SINGLE JOB
5406	T46	*TEST THE LPA SYSTEM USER MICRO-CODE'S ABILITY TO TAKE RANDOM SAMPLES SINGLE JOB
5495	T47	*TEST THAT MULTI JOBS CAN BE STARTED AND RUN
5693	T50	*HIGH SPEED A/D SAMPLE TEST (DEDICATED MODE, SPECIAL TEST
5768	T51	END OF TESTS
5773		END OF PASS ROUTINE
5834		BINARY TO OCTAL (ASCII) AND TYPE
5911		CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
5978		ERROR HANDLER ROUTINE
6030		ERROR MESSAGE TIMEOUT ROUTINE
6077		SCOPE HANDLER ROUTINE
6143		TTY INPUT ROUTINE
6285		TYPE ROUTINE
6364		READ AN OCTAL NUMBER FROM THE TTY
6402		APT COMMUNICATIONS ROUTINE
6459		POWER DOWN AND UP ROUTINES
6503		WIR ROUTINE TO WAIT FOR "DRLPX0" MICROCODE TO BECOME READY.
7588		TRAP DECODER
7611		TRAP TABLE
7673		ASCII MESSAGES.
7962	OMDT	DEDICATED MODE DISPATCH TABLE

30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83

```
;<BADGER>LPD.P11.54, 1-AUG-77 13:16:41, EDIT BY BADGER
;TITLE MAINDEC -11- DRLPA-A
;*COPYRIGHT (C) 1977
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY EDWARD C. BADGER, GREGORY GLEZMAN
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
;*
```

000001

\$TN=1

.SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<7:0>

.SBTTL ENABL GBL
TRAP CATCHER

000000

```
.=0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
```

000174	000000		
000176	000000		
000100	000104	000200	000002
000200	000137	002016	
000220	000137	034546	
000240	000137	041324	

```
.=174
DISPREG: .WORD 0 ;;SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER
.=100
.WORD 104,200,2
.=200
JMP START
.=220
JMP MAKEI
.=240
JMP RBUFR
```

.SBTTL BASIC DEFINITIONS

001100

```
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
```

000011

```
;*MISCELLANEOUS DEFINITIONS
HT= 11 ;;CODE FOR HORIZONTAL TAB
```

DRLPA.P11 BASIC DEFINITIONS

84	000012	LF=	12	::	CODE FOR LINE FEED
85	000015	CR=	15	::	CODE FOR CARRIAGE RETURN
86	000200	CRLF=	200	::	CODE FOR CARRIAGE RETURN-LINE FEED
87	177776	PS=	177776	::	PROCESSOR STATUS WORD
88		.EQUIV	PS,PSW		
89	177774	STKLMT=	177774	::	STACK LIMIT REGISTER
90	177772	PIRQ=	177772	::	PROGRAM INTERRUPT REQUEST REGISTER
91	177570	DSWR=	177570	::	HARDWARE SWITCH REGISTER
92	177570	DDISP=	177570	::	HARDWARE DISPLAY REGISTER
93					
94					
95	000000	:*GENERAL PURPOSE REGISTER DEFINITIONS			
96	000001	R0=	%0	::	GENERAL REGISTER
97	000002	R1=	%1	::	GENERAL REGISTER
98	000003	R2=	%2	::	GENERAL REGISTER
99	000004	R3=	%3	::	GENERAL REGISTER
100	000005	R4=	%4	::	GENERAL REGISTER
101	000006	R5=	%5	::	GENERAL REGISTER
102	000007	R6=	%6	::	GENERAL REGISTER
103	000006	R7=	%7	::	GENERAL REGISTER
104	000007	SP=	%6	::	STACK POINTER
105		PC=	%7	::	PROGRAM COUNTER
106					
107	000000	:*PRIORITY LEVEL DEFINITIONS			
108	000040	PR0=	0	::	PRIORITY LEVEL 0
109	000100	PR1=	40	::	PRIORITY LEVEL 1
110	000140	PR2=	100	::	PRIORITY LEVEL 2
111	000200	PR3=	140	::	PRIORITY LEVEL 3
112	000240	PR4=	200	::	PRIORITY LEVEL 4
113	000300	PR5=	240	::	PRIORITY LEVEL 5
114	000340	PR6=	300	::	PRIORITY LEVEL 6
115		PR7=	340	::	PRIORITY LEVEL 7
116					
117	100000	:*"SWITCH REGISTER" SWITCH DEFINITIONS			
118	040000	SW15=	100000		
119	020000	SW14=	40000		
120	010000	SW13=	20000		
121	004000	SW12=	10000		
122	002000	SW11=	4000		
123	001000	SW10=	2000		
124	000400	SW09=	1000		
125	000200	SW08=	400		
126	000100	SW07=	200		
127	000040	SW06=	100		
128	000020	SW05=	40		
129	000010	SW04=	20		
130	000004	SW03=	10		
131	000002	SW02=	4		
132	000001	SW01=	2		
133		SW00=	1		
134		.EQUIV	SW09,SW9		
135		.EQUIV	SW08,SW8		
136		.EQUIV	SW07,SW7		
137		.EQUIV	SW06,SW6		
		.EQUIV	SW05,SW5		

138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

000004
000010
000014
000014
000014
000014
000020
000024
000030
000034
000060
000064
000240
170460
000300
000001

```
.EQUIV SW04,SW4
.EQUIV SW03,SW3
.EQUIV SW02,SW2
.EQUIV SW01,SW1
.EQUIV SW00,SW0

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)
BIT15= 100000
BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

.*BASIC "CPU" TRAP VECTOR ADDRESSES
ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC=14 ;: "T" BIT
TRTVEC= 14 ;: TRACE TRAP
BPTVEC= 14 ;: BREAKPOINT TRAP (BPT)
IOTVEC= 20 ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24 ;: POWER FAIL
EMTVEC= 30 ;: EMULATOR TRAP (EMT) **ERROR**
TRAPVEC=34 ;: "TRAP" TRAP
TKVEC= 60 ;: TTY KEYBOARD VECTOR
TPVEC= 64 ;: TTY PRINTER VECTOR
PIRQVEC=240 ;: PROGRAM INTERRUPT REQUEST VECTOR
ABASE= 170460
AVECT1= 300
$TN=1
.SBTTL ACT11 HOOKS

:*****
:HOOKS REQUIRED BY ACT11
```

```

192          000244          $SVPC=.          ;SAVE PC
193          000046          .=46
194 000046 030772          $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
195          000052          .=52
196          000000          .WORD 0          ;;2)SET LOC.52 TO zERO
197          000244          .= $SVPC          ;; RESTORE PC
198
199          C~1000          .=1000
200          .SBTTL APT PARAMETER BLOCK
201
202          ;*****
203          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
204          ;*****
205          001000          .SX=.          ;;SAVE CURRENT LOCATION
206          000024          =24          ;;SET POWER FAIL TO pOINT TO START OF PROGRAM
207 000024 000200          200          ;;FOR APT START UP
208          000044          =44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
209 000044 001000          $APTHDR          ;;POINT TO APT HEADER BLOCK
210          001000          .=.SX          ;;RESET LOCATION COUNTER
211          ;*****
212          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
213          ;INTERFACE SPEC.
214
215 001000          $APTHD:
216 001000 000000          $HIBTS: .WORD 0          ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
217 001002 001174          $MADR: .WORD $MAIL          ;; ADDRESS OF APT MAILBOX (BITS 0-15)
218 001004 000002          $STMT: .WORD 2          ;; RUN TIM OF LONGEST TEST
219 001006 000170          $PASTM: .WORD 120.          ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
220 001010 000170          $UNITM: .WORD 120.          ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
221 001012 000031          .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)

```


222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275

.SBTTL COMMON TAGS

; THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
; USED IN THE PROGRAM.

001100
001100 000000
001102 000
001103 000
001104 000000
001106 000000
001110 000000
001112 000000
001114 000
001115 001
001116 000000
001120 000000
001122 000000
001124 000000
001126 000000
001130 000000
001132 000000
001134 000
001135 000
001136 000000
001140 177570
001142 177570
001144 177560
001146 177562
001150 177564
001152 177566
001154 000
001155 002
001156 012
001157 000
001160 000000
001162 000000
001164 177607
001170 077
001171 015
001172 000012

000377

SCMTAG: . =1100
; START OF COMMON TAGS
\$STNM: .WORD 0 ; CONTAINS THE TEST NUMBER
\$ERFLG: .BYTE 0 ; CONTAINS ERROR FLAG
\$ICNT: .WORD 0 ; CONTAINS SUBTEST ITERATION COUNT
\$LPADR: .WORD 0 ; CONTAINS SCOPE LOOP ADDRESS
\$LPERR: .WORD 0 ; CONTAINS SCOPE RETURN FOR ERRORS
\$ERTTL: .WORD 0 ; CONTAINS TOTAL ERRORS DETECTED
\$ITEMB: .BYTE 0 ; CONTAINS ITEM CONTROL BYTE
\$ERMAX: .BYTE 1 ; CONTAINS MAX. ERRORS PER TEST
\$ERRPC: .WORD 0 ; CONTAINS PC OF LAST ERROR INSTRUCTION
\$GDADR: .WORD 0 ; CONTAINS ADDRESS OF 'GOOD' DATA
\$BDADR: .WORD 0 ; CONTAINS ADDRESS OF 'BAD' DATA
\$GDDAT: .WORD 0 ; CONTAINS 'GOOD' DATA
\$BDDAT: .WORD 0 ; CONTAINS 'BAD' DATA
; RESERVED--NOT TO BE USED
\$AUTOB: .BYTE 0 ; AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 0 ; INTERRUPT MODE INDICATOR
\$SWR: .WORD DSWR ; ADDRESS OF SWITCH REGISTER
\$DISPLAY: .WORD DDISP ; ADDRESS OF DISPLAY REGISTER
\$TKS: 177560 ; TTY KBD STATUS
\$TKB: 177562 ; TTY KBD BUFFER
\$TPS: 177564 ; TTY PRINTER STATUS REG. ADDRESS
\$TPB: 177566 ; TTY PRINTER BUFFER REG. ADDRESS
\$NULL: .BYTE 0 ; CONTAINS NULL CHARACTER FOR FILLS
\$FILLS: .BYTE 2 ; CONTAINS # OF FILLER CHARACTERS REQUIRED
\$FILLC: .BYTE 12 ; INSERT FILL CHARS. AFTER A "LINE FEED"
\$TPFLG: .BYTE 0 ; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
\$TIMES: 0 ; MAX. NUMBER OF ITERATIONS
\$ESCAPE: 0 ; ESCAPE ON ERROR ADDRESS
\$BELL: .ASCIZ <207><377><377> ; CODE FOR BELL
\$QUES: .ASCII /?/ ; QUESTION MARK
\$CRLF: .ASCII <15> ; CARRIAGE RETURN
\$LF: .ASCIZ <12> ; LINE FEED

.SBTTL APT MAILBOX-ETABLE

.EVEN
\$MAIL: ; APT MAILBOX
\$MSGTY: .WORD AMSQTY ; MESSAGE TYPE CODE
\$FATAL: .WORD AFATAL ; FATAL ERROR NUMBER
\$TESTN: .WORD ATESTN ; TEST NUMBER
\$PASS: .WORD APASS ; PASS COUNT
\$DEVCT: .WORD ADEVCT ; DEVICE COUNT

276	001206	000000	\$UNIT:	.WORD	AUNIT	:: I/O UNIT NUMBER
277	001210	000000	\$MSGAD:	.WORD	AMSGAD	:: MESSAGE ADDRESS
278	001212	000000	\$MSGLG:	.WORD	AMSGLG	:: MESSAGE LENGTH
279	001214		\$ETABLE:			:: APT ENVIRONMENT TABLE
280	001214	000	\$ENV:	.BYTE	AENV	:: ENVIRONMENT BYTE
281	001215	000	\$ENVM:	.BYTE	AENVM	:: ENVIRONMENT MODE BITS
282	001216	000000	\$SWREG:	.WORD	ASWREG	:: APT SWITCH REGISTER
283	001220	000000	\$USWR:	.WORD	AUSWR	:: USER SWITCHES
284	001222	000000	\$CPUOP:	.WORD	ACPUOP	:: CPU TYPE, OPTIONS
285			*			BITS 15-11=CPU TYPE
286			*			11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
287			*			11/70=06, PDQ=07, Q=10
288			*			BIT 10=REAL TIME CLOCK
289			*			BIT 9=FLOATING POINT PROCESSOR
290			*			BIT 8=MEMORY MANAGEMENT
291	001224	000	\$MAMS1:	.BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
292	001225	000	\$MTYP1:	.BYTE	AMTYP1	:: MEM. TYPE, BLK#1
293			*			MEM. TYPE BYTE -- (HIGH BYTE)
294			*			900 NSEC CORE=001
295			*			300 NSEC BIPOLAR=002
296			*			500 NSEC MOS=003
297	001226	000000	\$MADR1:	.WORD	AMADR1	:: HIGH ADDRESS, BLK#1
298			*			MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
299	001230	000	\$MAMS2:	.BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
300	001231	000	\$MTYP2:	.BYTE	AMTYP2	:: MEM. TYPE, BLK#2
301	001232	000000	\$MADR2:	.WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
302	001234	000	\$MAMS3:	.BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
303	001235	000	\$MTYP3:	.BYTE	AMTYP3	:: MEM. TYPE, BLK#3
304	001236	000000	\$MADR3:	.WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
305	001240	000	\$MAMS4:	.BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
306	001241	000	\$MTYP4:	.BYTE	AMTYP4	:: MEM. TYPE, BLK#4
307	001242	000000	\$MADR4:	.WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
308	001244	000300	\$VECT1:	.WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
309	001246	000000	\$VECT2:	.WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
310	001250	170460	\$BASE:	.WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
311	001252	000000	\$DEVM:	.WORD	ADEVM	:: DEVICE MAP
312	001254	000000	\$CDW1:	.WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
313	001256		\$ETEND:			
314			.MEXIT			

315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ;:POINTS TO THE ERROR MESSAGE
;* DH ;:POINTS TO THE DATA HEADER
;* DT ;:POINTS TO THE DATA
;* DF ;:POINTS TO THE DATA FORMAT

001256

\$ERRTB:

;ITEM 1

001256 041504
001260 042240
001262 042516
001264 042606

EM1
DH1
DT1
DF0

;LPA-11 ADDRESS ERROR
;ERRPC ADDRESS
;\$ERRPC,KMADO
;ALL NUMBER ARE IN OCTAL FORM.

;ITEM 2

001266 041527
001270 042261
001272 042524
001274 042606

EM2
DH2
DT2
DF0

;LPA(KMC-11) DATA ERROR
;ERRPC EXP'D REC'D
;\$ERRPC,\$GDDAT,\$BDDAT
;ALL NUMBER ARE IN OCTAL FORM.

;ITEM 3

001276 041560
001300 042312
001302 042534
001304 042606

EM3
DH3
DT3
DF0

;LPA (KMC-11) INSTRUCTION ERROR
;ERRPC
;\$ERRPC
;ALL NUMBER ARE IN OCTAL FORM.

;ITEM 4

001306 041620
001310 042312
001312 042534
001314 042606

EM4
DH3
DT3
DF0

;LPA (M8254) INIT.ERROR
;ERRPC
;\$ERRPC
;ALL NUMBER ARE IN OCTAL FORM.

;ITEM 5

001316 041620
001320 042261
001322 042524
001324 042606

EM4
DH2
DT2
DF0

;LPA (M8254) INIT.ERROR
;ERRPC EXP'ED REC'ED
;\$ERRPC,\$GDDAT,\$BDDAT
;ALL NUMBER ARE IN OCTAL FORM.

;ITEM 6

369	001326	041650	EM6	:LPA (M8254) SILO DATA ERROR
370	001330	042261	DH2	:ERRPC EXP'ED REC'ED
371	001332	042524	DT2	:\$ERRPC,\$GDDAT,\$BDDAT
372	001334	042606	DF0	:ALL NUMBER ARE IN OCTAL FORM.
373				
374			;ITEM 7	
375				
376	001336	041705	EM7	:LPA (M8254) FAST PATH DATA ERROR
377	001340	042261	DH2	:ERRPC EXP'ED REC'ED
378	001342	042524	DT2	:\$ERRPC,\$GDDAT,\$BDDAT
379	001344	042606	DF0	:ALL NUMBER ARE IN OCTAL FORM.
380				
381			;ITEM 10	
382				
383	001346	041747	EM10	:LPA (AD11K) CSR ERROR
384	001350	042261	DH2	:ERRPC EXP'ED REC'ED
385	001352	042524	DT2	:\$ERRPC,\$GDDAT,\$BDDAT
386	001354	042606	DF0	:ALL NUMBER ARE IN OCTAL FORM.
387				
388			;ITEM 11	
389				
390	001356	041776	EM11	:LPA (KW11K) CSR ERROR
391	001360	042261	DH2	:ERRPC EXP'ED REC'ED
392	001362	042524	DT2	:\$ERRPC,\$GDDAT,\$BDDAT
393	001364	042606	DF0	:ALL NUMBER ARE IN OCTAL FORM.
394				
395			;ITEM 12	
396				
397	001366	042025	EM12	:LPA (DR11K) ERROR
398	001370	042261	DH2	:ERRPC EXP'ED REC'ED
399	001372	042524	DT2	:\$ERRPC,\$GDDAT,\$BDDAT
400	001374	042606	DF0	:ALL NUMBER ARE IN OCTAL FORM.
401				
402			;ITEM 13	
403				
404	001376	000000	0	:NO-ERROR 13 (BAD LUCK!)
405	001400	000000	0	
406	001402	000000	0	
407	001404	000000	0	
408				
409			;ITEM 14	
410				
411	001406	042050	EM14	:LPA (AA-11K) ERROR
412	001410	042261	DH2	:ERRPC EXP'ED REC'ED
413	001412	042524	DT2	:\$ERRPC,\$GDDAT,\$BDDAT
414	001414	042606	DF0	:ALL NUMBER ARE IN OCTAL FORM.
415				
416			;ITEM 15	
417				
418	001416	042074	EM15	:LPA-11 (AR11K) ERROR
419	001420	042261	DH2	:ERRPC EXP'ED REC'ED
420	001422	042524	DT2	:\$ERRPC,\$GDDAT,\$BDDAT
421	001424	042606	DF0	:ALL NUMBER ARE IN OCTAL FORM.
422				

```

423 ;ITEM 16
424
425 001426 042116 EM16 ;LPA-11 (LPS-11) ERROR
426 001430 042261 OH2
427 001432 042524 DT2
428 001434 042606 DFO ;ALL NUMBER ARE IN OCTAL FORM.
429
430 ;ITEM 17
431
432 001436 042142 EM17 ;LPA-11 FUNCTIONAL ERROR
433 001440 042321 DH17
434 001442 042540 DT17
435 001444 042606 DFO ;ALL NUMBER ARE IN OCTAL FORM.
436
437 ;ITEM 20
438
439 001446 042173 EM20
440 001450 042361 DH20
441 001452 042552 DT20
442 001454 042606 DFO ;ALL NUMBER ARE IN OCTAL FORM.
443
444 ;ITEM 21
445 001456 042142 EM17 ;LPA FUNCTIONAL ERROR
446 001460 042427 DH21 ;ERRPC TSTNO USJNO USRDA ALPCO ALPCI ALPSO
447 001462 042566 DT21 ;ERRPC,STE:TN,USJNO,USRDA,ALPCO,ALPCI,APSO
448 001464 042606 DFO ;ALL NUMBER ARE IN OCTAL FORM.
449
450 ;DEVICE ADDRESSES AS USED BY PROGRAM.
451 ;DO NOT "PATCH" THESE LOCATIONS
452 ;LOCATIONS ARE ALTERED BY PROGRAM BASED ON YOUR INFO IN $BASE
453
454 001466 KMCSR:
455
456 ;ADDRESS OF KMC-11 OF LPA-11 THE ADDR FOR KMADO MAY BE
457 ; CHANGED BY THE USER TO REFLECT
458 ; A DIFFERENT KMC-11 ADDR. THE
459 ; REST OF THE ADDRESSES WILL
460 ; BE CHANGED BY THE PROGRAM.
461
462
463 001466 LPCI:
464 001466 170460 KMADO: .WORD ABASE ;BASE KMC ADDR. MAY BE PATCHED BY USER.
465
466 001470 LPMR:
467 001470 170461 KMAD1: .WORD ABASE+1 ;>DO NOT <;KMC-CSR ADDR
468 001472 LPCO:
469 001472 170462 KMAD2: .WORD ABASE+2 ;>PATCH <;
470 001474 LPSO:
471 001474 170463 KMAD3: .WORD ABASE+3 ;>THIS AREA <
472 001476 LPADL:
473 001476 170464 KMAD4: .WORD ABASE+4 ;
474 001500 LPADH:
475 001500 170465 KMAD5: .WORD ABASE+5 ;>DO NOT <
476 001502 LPMS1:

```

```

477 001502 170466 KMAD6: .WORD ABASE+6 ;>PATCH <
478 001504 LPMS2:
479 001504 170467 KMAD7: .WORD ABASE+7 ;>THIS AREA <
480
481 001506 000300 VECTOR: .WORD AVECT1&777 ;BASE VECTOR OF KMC
482 001510 000304 VECTPS: .WORD 4+AVECT1&777 ;VECTR ADDR.+2
483
484 001512 000004 VERSN: .WORD 4 ;CURRENT VERSION NUMBER OF MICROCODE.
485
486 001514 000000 .DVLS: .WORD 0 ;/DEVICE LIST OF I/O ADDR. DEFINED
487 001516 000020 .BLKW 16. ;/BY INIT.
488
489
490

```

```

491 001556 000300 VECT1: .WORD AVECT1
492
493 001560 000304 VECT2: .WORD AVECT1+4
494
495 .SBTTL *****
496 .SBTTL SR1 DEVICE PRESENT REG
497 .SBTTL *****
498
499

```

```

500 .REM %
501
502 SR1 INFORMS THIS DIAGNOSTIC AS TO WHAT DEVICES
503 ARE ON THE I/O BUS.
504 SR1 DEFAULTS TO 1 KW11K AND 1 AD11K.
505 IF YOUR CONFIGURATION IS DIFFERENT, YOU MUST CHANGE
506 THIS LOCATION

```

WORD BIT=1	OCTAL	DEVICE
0	000001	1ST AD11K
1	000002	1ST KW11K
2	000004	1ST DR11K
3	000010	1ST AR11K
4	000020	2ND AD11K#2
5	000040	2ND DR11K
6	000100	RESERVED
7	000200	3RD DR11K
8	000400	4TH DR11K
9	001000	5TH DR11K
10	002000	AR11
11	004000	RESERVED
12	010000	LPSAD (LPS A/D)
13	020000	LPSKW (LPS REAL TIME CLOCK)
14	040000	LPSVC (LPS D/A)
15	100000	LPSDR (LPS DIGITAL J/O)

```

507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529 001562 000003 SR1: .WORD 003 ;DEVICE PRESENT FLAG DEFAULT
530 ;IS 1 KW11K, 1 AD11K

```


531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584

.SBTTL *****
.SBTTL SR2 DEVICE SET-UP REG
.SBTTL *****

.REM %

SR2 INFORMS THIS DIAGNOSTIC AS TO HOW THE
DEVICES SELECTED BY SR1 ARE SET-UP FOR TEST.
WHILE NO SPECIAL SETUP IS REQUIRED TO RUN THIS TEST,
THE DEPTH OF COVERAGE WILL INCREASE IF SPECIAL SETUPS
ARE PERFORMED.

WORD BIT=1	OCTAL	FUNCTION
0	000001	AD11K HAS G5034 WRAP-AROUND MODULE.
2	000004	DR11K #1 HAS LOOP BACK CABLE
3	000010	AR11K HAS SCOPE DISPLAY (VISUAL TEST)
4	000020	AD11K #2 HAS G5034 WRAP AROUND MODULE.
5	000040	DR11K #2 HAS LOOPBACK CABLE
7	000200	DR11K #3 HAS LOOP BACK CABLE.
8	000400	DR11K #4 HAS LOOP BACK CABLE.
9	001000	DR11K #5 HAS LOOP BACK CABLE.
10	002000	AR11 HAS G5034 WRAP AROUND MODULE.
11	004000	AR11 HAS SCOPE DISPLAY (VISUAL TEST)
14	040000	LPS-11 D/A HAS SCOPE DISPLAY (VISUAL TEST)
15	100000	LPS-11 DIGITAL I/O HAS LOOP BACK CABLE

%

001564 000000

SR2: .WORD 0

.SBTTL *****
.SBTTL LPA-11 I/O BUS DEVICE ADDRESS DEFINITION AND LIST
.SBTTL *****

.REM %
THE FOLLOWING ARE A LIST OF DEVICE ADDRESSES. YOU MAY
CHANGE THE ADDRESS OF A DEVICE ONLY BY MODIFYING THE FOLLOWING LIST.
DO NOT CHANGE ANY OTHER ADDRESSES!
%

001566	170400	AD11K: .WORD	170400	;AD11K ADDRESS.
001570	170404	KW11K: .WORD	170404	;KW11K ADDRESS.
001572	167770	DR11K1: .WORD	167770	;DR11K #1 ADDR.
001574	170416	AR11K: .WORD	170416	;AR11 ADDRESS.
001576	170440	AD11K2: .WORD	170440	;AD11K #2 ADDRESS.
001600	167760	DR11K2: .WORD	167760	;DR11K #2 ADDRESS.
001602	167750	DR11K3: .WORD	167750	;DR11K #3 ADDRESS.
001604	167740	DR11K4: .WORD	167740	;DR11K #4 ADDRESS.
001606	167730	DR11K5: .WORD	167730	;DR11K #5 ADDRESS.
001610	170400	AR11K: .WORD	170400	;AR11 ADDRESS.
001612	170400	LPS11: .WORD	170400	;LPS11 BASE ADDRESS.

```

585 .REM $
586 $ END OF PATCHABLE DEVICE ADDRESSES.
587 $
588 .SBTTL AD11-K ADDRESSES
589
590
591 001614 170400 STREG1: .WORD 170400 ;CONTROL AND STATUS REGISTER.
592
593 001616 ADBUF1: ;ADDRESS OF BUFFER REG. AND
594 001616 170402 ADAC: .WORD 170402 ;DIGITAL OUTPUT REG.
595 001620 170440 STREG2: .WORD 170440 ;AD11K #2 CSR
596 001622 ADBUF2:
597 001622 170442 ADAC2: .WORD 170442 ;AD11K #2 OUTPUT/INPUT
598 .SBTTL AA11-K ADDRESSES
599
600 001624 170416 AACSR: .WORD 170416 ;AA11K CSR
601 001626 170420 DAC0: .WORD 170420
602 001630 170422 DAC1: .WORD 170422
603 001632 170424 DAC2: .WORD 170424
604 001634 170426 DAC3: .WORD 170426
605
606
607 .SBTTL DR11K #1 ADDRESS
608 001636 167770 DRCR1: .WORD 167770 ;DR11K #1 CSR.
609 001640 167774 DROA1: .WORD 167770+4 ;DR11K OUTPUT REG.
610 001642 167772 DRIA1: .WORD 167770+2 ;DR11K INPUT REG.
611
612 .SBTTL DR11K #2 ADDRESS
613 001644 167760 DRCR2: .WORD 167760 ;DR11K #2 CSR.
614 001646 167764 DROA2: .WORD 167760+4 ;DR11K OUTPUT REG.
615 001650 167762 DRIA2: .WORD 167760+2 ;DR11K INPUT REG.
616
617 .SBTTL DR11K #3 ADDRESS
618 001652 167750 DRCR3: .WORD 167750 ;DR11K #3 CSR.
619 001654 167754 DROA3: .WORD 167750+4 ;DR11K OUTPUT REG.
620 001656 167752 DRIA3: .WORD 167750+2 ;DR11K INPUT REG.
621
622 .SBTTL DR11K #4 ADDRESS
623 001660 167740 DRCR4: .WORD 167740 ;DR11K #4 CSR.
624 001662 167744 DROA4: .WORD 167740+4 ;DR11K OUTPUT REG.
625 001664 167742 DRIA4: .WORD 167740+2 ;DR11K INPUT REG.
626
627 .SBTTL DR11K #5 ADDRESS
628 001666 167730 DRCR5: .WORD 167730 ;DR11K #5 CSR.
629 001670 167734 DROA5: .WORD 167730+4 ;DR11K OUTPUT REG.
630 001672 167732 DRIA5: .WORD 167730+2 ;DR11K INPUT REG.
631
632
633 .SBTTL KW11K ADDRESS
634
635
636 001674 170404 KWADPO: .WORD 170404 ;CLOCK A CSR
637
638 001676 170406 KWADR1: .WORD 170406 ;CLOCK A PRESENT REG.

```

```

639
640 001700 170430 KWADR2: .WORD 170430 ;CLOCK A COUNTER REG.
641
642 001702 170432 KWADR4: .WORD 170432 ;CLOCK B CSR
643
644 001704 170434 KWADR5: .WORD 170434 ;CLOCK B PRESENT REG.
645
646 001706 170436 KWADR6: .WORD 170436 ;CLOCK B COUNT REG.
647
648 .SBTTL AR11 ADDRESSES
649
650
651 001710 170400 ARADS: .WORD 170400 ;A/D STATUS REG
652 001712 170402 ARADB: .WORD 170402 ;A/D BUFFER
653 001714 170404 ARCS: .WORD 170404 ;CLOCK STATUS
654 001716 170406 ARCB: .WORD 170406 ;CLOCK BUFFER/PRESET
655 001720 170410 ARDS: .WORD 170410 ;DISPLAY STATUS
656 001722 170412 ARXB: .WORD 170412 ;X BUFFER
657 001724 170414 ARYB: .WORD 170414 ;Y BUFFER
658 001726 170416 ARCC: .WORD 170416 ;CLOCK COUNT REG.
659
660
661 .SBTTL LPS-11 ADDRESSES
662
663 001730 170400 LPADSR: .WORD 170400 ;A/D STATUS REG.
664 001732 170402 LPADBR: .WORD 170402 ;A/D BUFFER REG.
665 001734 170404 LPCKSR: .WORD 170404 ;CLOCK STATUS REG.
666 001736 170406 LPCKBR: .WORD 170406 ;CLOCK BUFFER REG.
667 001740 170410 LPIOSR: .WORD 170410 ;I/O STATUS REG.
668 001742 170412 LPIOIR: .WORD 170412 ;I/O INPUT REG.
669 001744 170414 LPIOOR: .WORD 170414 ;I/O OUTPUT REG.
670 001746 170416 LPDASR: .WORD 170416 ;D/A STATUS REG.
671 001750 170420 LPDAXR: .WORD 170420 ;D/A X REG.
672 001752 170422 LPDAYR: .WORD 170422 ;D/A Y REG.
673
674 ;MISC. STORAGE LOCATIONS
675
676 001754 000000 MYTEMP: .WORD 0 ;TEMP STORAGE
677 001756 000000 USJNO: .WORD 0 ;USERS JOB NUMBER (IN R FLASH)
678 001760 000000 USRDA: .WORD 0 ;USERS RDA ADDR.
679 001762 000000 ALPCI: .WORD 0 ;CONTENTS OF CONTROL IN REG.
680 001764 000000 ALPCO: .WORD 0 ;CONTENTS OF CONTROL OUT REG.
681 001766 000000 ALPSO: .WORD 0 ;CONTENTS OF HIGH BYT OF ABOVE. (STATUS REG)
682 001770 000000 ALPADL: .WORD 0 ;CONTENTS OF ADDR. REG.
683 001772 000 $VERSN: .BYTE 0,3 ;CONTAIN UCODE SYMBOLS
684 001774 000000 TAXING: .WORD 0
685 001776 000000 ERCNT: .WORD 0 ;NO OF ERRORS.
686
687 ;TO BE KW11P ADDR. OR NO ADDR. AT ALL
688
689
690 000073 STPC00=73
691 000074 LOPC=74
692 000007 VBM=7
    
```

003

```

693      000007
694 002000 177546      KW11L:  USWH=7      ; ADDR. OF KW11L
695 002002 172540      KW11P:  .WORD      172540 ; ADDR. OF KW11P.
696
697 002004 002006      NOCLK:  .WORD      CLKTMP ; ADDR. OF NO CLOCK.
698 002006 000000      CLKTMP: .WORD      0
699 002010 000100      KWVECL: .WORD      100 ; VECOTR OF KW11L
700 002012 000104      KWVECP: .WORD      104 ; VECTOR OF KW11L
701 002014 000000      FUDGE:  .WORD      0 ; FUDGE FACTOR
702
703
704 002016
705      .SBTTL INITIALIZE THE COMMON TAGS
706      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
707 002016 012706 001100      MOV      $CMTAG,R6 ; FIRST LOCATION TO BE CLEARED
708 002022 005026      CLR      (R6)+ ; CLEAR MEMORY LOCATION
709 002024 022706 001140      CMP      $SWR,R6 ;; DONE?
710 002030 001374      BNE      -6 ; LOOP BACK IF NO
711 002032 012706 001100      MOV      $STACK,SP ; SETUP THE STACK POINTER
712      ;;INITIALIZE A FEW VECTORS
713 002036 012737 032014 000020      MOV      $SCOPE,$IOTVEC ; IOT VECTOR FOR SCOPE ROUTINE
714 002044 012737 000340 000022      MOV      $340,$IOTVEC+2 ; LEVEL 7
715 002052 012737 031464 000030      MOV      $ERROR,$EMTVEC ; EMT VECTOR FOR ERROR ROUTINE
716 002060 012737 000340 000032      MOV      $340,$EMTVEC+2 ; LEVEL 7
717 002066 012737 041242 000034      MOV      $TRAP,$TRAPVEC ; TRAP VECTOR FOR TRAP CALLS
718 002074 012737 000340 000036      MOV      $340,$TRAPVEC+2; LEVEL 7
719 002102 012737 033704 000024      MOV      $PWARN,$PWAVEC ; POWER FAILURE VECTOR
720 002110 012737 000340 000026      MOV      $340,$PWAVEC+2 ; LEVEL 7
721 002116 005037 001160      CLR      $TIMES ; INITIALIZE NUMBER OF ITERATIONS
722 002122 005037 001162      CLR      $ESCAPE ; CLEAR THE ESCAPE ON ERROR ADDRESS
723 002126 112737 000001 001115      MOVB    #1,$SERMAX ; ALLOW ONE ERROR PER TEST
724 002134 012737 002134 001106      MOV      $,$SLPADR ; INITIALIZE THE LOOP ADDRESS FOR SCOPE
725 002142 012737 002142 001110      MOV      $,$SLPERR ; SETUP THE ERROR LOOP ADDRESS
726      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
727      ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
728 002150 013746 000004      MOV      $ERRVEC,-(SP) ; SAVE ERROR VECTOR
729 002154 012737 002210 000004      MOV      $64,$ERRVEC ; SET UP ERROR VECTOR
730 002162 012737 177570 001140      MOV      $DSWR,$SWR ; SETUP FOR A HARDWARE SWICH REGISTER
731 002170 012737 177570 001142      MOV      $DISP,$DISPLAY ; AND A HARDWARE DISPLAY REGISTER
732 002176 022777 177777 176734      CMP      #-1,$SWR ; TRY TO REFERENCE HARDWARE SWR
733 002204 001012      BNE      65$ ; BRANCH IF NO TIMEOUT TRAP OCCURRED
734      ; AND THE HARDWARE SWR IS NOT = -1
735 002206 000403      BR      65$ ; BRANCH IF NO TIMEOUT
736 002210 012716 002216 64$: MOV      $65$,(SP) ; SET UP FOR TRAP RETURN
737 002214 000002      RTI
738 002216 012737 000176 001140 65$: MOV      $SWREG,$SWR ; POINT TO SOFTWARE SWR
739 002224 012737 000174 001142      MOV      $DISPREG,$DISPLAY
740 002232 012637 000004 66$: MOV      (SP)+,$ERRVEC ; RESTORE ERROR VECTOR
741
742 002236 005037 001202      CLR      $PASS ; CLEAR PASS COUNT
743 002242 132737 000200 001215      BITB    $APTSIZE,$ENVM ; TEST USER SIZE UNDER APT
744 002250 001403      BEQ     67$ ; YES,USE NON-APT SWITCH
745 002252 012737 001216 001140      MOV     $SSWREG,$SWR ; NO,USE APT SWITCH REGISTER
746 002260

```

```

747
748 002260 013700 001250      MOV      $BASE,RO      ;GET BASE KMC-ADDR.
749 002264 012701 001466      MOV      #KMADD,P1    ;STORAGE OF ADDR. LIST
750 002270 010021      1$: MOV      RO,(1)+    ;STORE ADDR.
751 002272 005200      INC      RO            ;UPDATE ADDR.
752 002274 020127 001506      CMP      R1,#KMAD7+2 ;DONE WHOLE LIST?
753 002300 001373      BNE      1$
754
755 002302 005037 001514      CLR      .DVLS
756
757 002306 013737 001244 001556      MOV      $VECT1,VECT1 ;GET VECTOR ADDR.
758 002314 042737 170000 001556      BIC      #170000,VECT1
759 002322 013737 001556 001560      MOV      VECT1,VECT2
760 002330 052737 000004 001560      BIS      #4,VECT2
761
762      ;THE FOLLOWING SECTION OF CODE SETS UP DEVICE ADDRESSES
763      ;BASED ON DEFAULT OR OPERATOR MODIFIED DATA IN LOCATIONS
764      ;"AD11K" THROUGH "LPS11"
765
766 002336 013700 001566      MOV      AD11K,RO      ;FIX AD11K'S ADDRESS
767 002342 010037 001614      MOV      RO,STREG1
768 002346 010037 001616      MOV      RO,ADAC
769 002352 062737 000002 001616      ADD      #2,ADAC
770
771 002360 013700 001570      MOV      KW11K,RO      ;FIX KW11K ADDRESS
772 002364 010037 001674      MOV      RO,KWADR0
773 002370 010037 001676      MOV      RO,KWADR1
774 002374 010037 001700      MOV      RO,KWADR2
775 002400 010037 001702      MOV      RO,KWADR4
776 002404 010037 001704      MOV      RO,KWADR5
777 002410 010037 001706      MOV      RO,KWADR6
778 002414 062737 000002 001676      ADD      #2,KWADR1
779 002422 052737 000024 001700      ADD      #24,KWADR2
780 002430 062737 000026 001702      ADD      #26,KWADR4
781 002436 062737 000030 001704      ADD      #30,KWADR5
782 002444 062737 000032 001706      ADD      #32,KWADR6
783
784
785
786 002452 013700 001572      MOV      DR11K1,RO      ;FIX DR11K #1 ADDRESS
787 002456 010037 001636      MOV      RO,DRCR1
788 002462 010037 001640      MOV      RO,DROA1
789 002466 010037 001642      MOV      RO,DRIA1
790 002472 062737 000004 001640      ADD      #4,DROA1
791 002500 062737 000002 001642      ADD      #2,DRIA1
792
793 002506 013700 001600      MOV      DR11K2,RO      ;FIX DR11K #2 ADDRESS
794 002512 010037 001644      MOV      RO,DRCR2
795 002516 010037 001646      MOV      RO,DROA2
796 002522 010037 001650      MOV      RO,DRIA2
797 002526 062737 000004 001646      ADD      #4,DROA2
798 002534 062737 000002 001650      ADD      #2,DRIA2
799
800 002542 013700 001602      MOV      DR11K3,RO      ;FIX DR11K #3 ADDRESS

```

801	002546	010037	001652		MOV	RO,DRCR3	
802	002552	010037	001654		MOV	RO,DROA3	
803	002556	010037	001656		MOV	RO,DRIA3	
804	002562	062737	000004	001654	ADD	#4,DROA3	
805	002570	062737	000002	001656	ADD	#2,DRIA3	
806							
807	002576	013700	001604		MOV	DR11K4,RO	;FIX DR11K #4 ADDRESS
808	002602	010037	001660		MOV	RO,DRCR4	
809	002606	010037	001662		MOV	RO,DROA4	
810	002612	010037	001664		MOV	RO,DRIA4	
811	002616	062737	000004	001662	ADD	#4,DROA4	
812	002624	062737	000002	001664	ADD	#2,DRIA4	
813							
814	002632	013700	001606		MOV	DR11K5,RO	;FIX DR11K #5 ADDRESS
815	002636	010037	001666		MOV	RO,DRCR5	
816	002642	010037	001670		MOV	RO,DROA5	
817	002646	010037	001672		MOV	RO,DRIA5	
818	002652	062737	000004	001670	ADD	#4,DROA5	
819	002660	062737	000002	001672	ADD	#2,DRIA5	
820							
821	002666	013700	001574		MOV	AR11K,RO	;FIX AR11K ADDRESS
822	002672	010037	001624		MOV	RO,ARCSR	
823	002676	010037	001626		MOV	RO,DACO	
824	002702	010037	001630		MOV	RO,DAC1	
825	002706	010037	001632		MOV	RO,DAC2	
826	002712	010037	001634		MOV	RO,DAC3	
827	002716	062737	000002	001626	ADD	#2,DACO	
828	002724	062737	000004	001630	ADD	#4,DAC1	
829	002732	062737	000006	001632	ADD	#6,DAC2	
830	002740	062737	000010	001634	ADD	#10,DAC3	
831							
832	002746	013700	001576		MOV	AD11K2,RO	;FIX SECOND AD11K'S ADDR.
833	002752	010037	001620		MOV	RO,STREG2	
834	002756	010037	001622		MOV	RO,ADAC2	
835	002762	062737	000002	001622	ADD	#2,ADAC2	
836							
837	002770	013700	001610		MOV	AR11K,RO	;FIX AR11 ADDR.
838	002774	010037	001710		MOV	RO,ARADS	
839	003000	010037	001712		MOV	RO,ARADB	
840	003004	010037	001714		MOV	RO,ARCS	
841	003010	010037	001716		MOV	RO,ARCB	
842	003014	010037	001720		MOV	RO,ARDS	
843	003020	010037	001722		MOV	RO,ARXB	
844	003024	010037	001724		MOV	RO,ARYB	
845	003030	010037	001726		MOV	RO,ARCC	
846	003034	062737	000002	001712	ADD	#2,ARADB	
847	003042	062737	000004	001714	ADD	#4,ARCS	
848	003050	062737	000006	001716	ADD	#6,ARCB	
849	003056	062737	000010	001720	ADD	#10,ARDS	
850	003064	062737	000012	001722	ADD	#12,ARXB	
851	003072	062737	000014	001724	ADD	#14,ARYB	
852	003100	062737	000016	001726	ADD	#16,ARCC	
853							
854	003106	013700	001612		MOV	LPS11,RO	;FIX LPS-11 ADDRESSES


```

855 003112 010037 001730      MOV      RO,LPADSR
856 003116 010037 001732      MOV      RO,LPADBR
857 003122 010037 001734      MOV      RO,LPCKSR
858 003126 010037 001736      MOV      RO,LPCKBR
859 003132 010037 001740      MOV      RO,LPIOSR
860 003136 010037 001742      MOV      RO,LPIOIR
861 003142 010037 001744      MOV      RO,LPIOOR
862 003146 010037 001746      MOV      RO,LPDASR
863 003152 010037 001750      MOV      RO,LPDAXR
864 003156 010037 001752      MOV      RO,LPDAYR
865 003162 062737 000002 001732      ADD      #2,LPADBR
866 003170 062737 000004 001734      ADD      #4,LPCKSR
867 003176 062737 000006 001736      ADD      #6,LPCKBR
868 003204 062737 000010 001740      ADD      #10,LPIOSR
869 003212 062737 000012 001742      ADD      #12,LPIOIR
870 003220 062737 000014 001744      ADD      #14,LPIOOR
871 003226 062737 000016 001746      ADD      #16,LPDASR
872 003234 062737 000020 001750      ADD      #20,LPDAXR
873 003242 062737 000022 001752      ADD      #22,LPDAYR
874 003250 005037 002014      CLR      FUDGE
875
876
877 003254 012737 003276 000004      MOV      #10$,ERRVEC      ;SET UP FOR TIME-OUT IF NO KW11L
878 003262 005777 176512      TST      @KW11L          ;TRAP HERE IF NO L
879 003266 013737 002000 040624      MOV      KW11L,RTCCSR    ;MUST BE HERE,RECORD ADDR.
880 003274 000414      BR
881 003276 012737 003320 000004 10$:      MOV      #20$,ERRVEC    ;SET UP FOR TIME-OUT IF NO KW11P
882 003304 005777 176472      TST      @KW11P          ;TRAP HERE IF NO P
883 003310 013737 002002 040624      MOV      KW11P,RTCCSR    ;MUST BE HERE,RECORD ADDR.
884 003316 000403      BR
885 003320 013737 002004 040624 20$:      MOV      NOCLK,RTCCSR    ;NO CLOCK! OHOH.
886 003326 012706 001100 50$:      MOV      #STACK,SP      ;RESTORE STACK
887 003332 012737 000006 000004      MOV      #6,ERRVEC      ;RESTORE ERROR VECTOR.
888
889 003340 012777 040616 176442      MOV      #CLKINT,@KWVECL ;SET UP CLOCK VECTOR.
890 003346 012777 040616 176436      MOV      #CLKINT,@KWVECP
891
892      .SBTTL  TYPE PROGRAM NAME
893      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
894      INC      #-1          ;;FIRST TIME?
895      BNE      68$          ;;BRANCH IF NO
896      TYPE      69$          ;;TYPE ASCIZ STRING
897      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
898      TST      @#42          ;;ARE WE RUNNING UNDER XXDP/ACT?
899      BNE      70$          ;;BRANCH IF YES
900      CMPB     $ENV,#1      ;;ARE WE RUNNING UNDER APT?
901      BEQ      70$          ;;BRANCH IF YES
902      CMP      SWR,#SWREG    ;;SOFTWARE SWITCH REG SELECTED?
903      BNE      71$          ;;BRANCH IF NO
904      GTSWR          ;;GET SOFT-SWR SETTINGS
905      BR      71$
906      MOVB     #1,$AUTOB     ;;SET AUTO-MODE INDICATOR
907      BR      68$
908      ;;69$: .ASCIZ <CRLF>#MD-11-DRLPA-A LPA-11 DIAGNOSTIC #<CRLF>

```

```

909 003474          68$:
910
911 003474 005037 001776          CLR      ERCNT
912 003500          RSTART:
913 003500 005037 001514          CLR      .DVLS
914
915          ;*****
916          ;*TEST 1          *TEST THE ADDRESSABILITY OF THE KMC-11
917          ;*****
918 003504 000240          †ST1:  NOP
919 003506 012737 000050 001160          MOV      #50,$TIMES          ;;DO 50 ITERATIONS
920 003514 012737 003552 001106          MOV      #1$, $LPADR          ;;SET SCOPE LOOP ADDRESS
921
922 003522 012737 000001 001102          MOV      #1,$STINM
923 003530 012737 000001 001200          MOV      #1,$TESTN
924 003536 012737 003552 001106          MOV      #1$, $LPADR
925 003544 012737 003552 001110          MOV      #1$, $LPERR
926
927 003552 013746 000004          1$:  MOV      ERRVEC, -(SP)          ;SAVE CONTENTS OF LOC 4.
928 003556 012737 003626 000004          MOV      #2$, ERRVEC          ;SET TIME-OUT TO ERROR REPORTER
929 003564 013746 000006          MOV      ERRVEC+2, -(SP)
930 003570 012737 000340 000006          MOV      #340, ERRVEC+2
931
932 003576 042777 000000 175662          BIC      #0, @KMADO          ;CAUSE A DATA IN, DATA OUT
933
934 003604 005077 175656          CLR      @KMADO          ;CLEAR KMC
935 003610 005077 175656          CLR      @KMAD2
936 003614 005077 175656          CLR      @KMAD4
937 003620 005077 175656          CLR      @KMAD6
938
939 003624 000403          BR       3$
940
941
942 003626 104001          2$:  ERROR  1          ;TIME-OUT TRAP TO LOC. 4 WHEN
943
944
945
946
947
948 003630 062706 000004          ADD      #4, SP          ;TAKE CARE OF TRAP SP.
949
950 003634 012637 000006          3$:  MOV      (SP)+, ERRVEC+2          ;RESTORE ERROR VECTOR.
951 003640 012637 000004          MOV      (SP)+, ERRVEC
952
953
954          ;*****
955          ;*TEST 2          *TEST KMC-11 FUNCTIONS
956
957
958          ;*
959          ;*IN THIS TEST WE ATTEMPT TO RUN AN INSTRUCTION TEST ON THE KMC-11.
960          ;*YOU WILL NOT BE ABLE TO LOOP ON AN INDIVIDUAL SUBTEST, BUT MUST LOOP ON
961          ;*THIS WHOLE TEST SINCE ONCE THE MICRO-CODE IS STARTED, IT EXECUTES IN-LINE
962          ;*WITH NO-LOOP BACK CAPABILITIES.
          ;*IF THIS TEST FAILS YOU SHOULD RUN THE REGULAR KMC-11 DIAGNOSTICS

```

```

963 ;*AVAILABLE FOR THE KMC-11
964 ;*FOR EACH TEST, WE WILL LIST THE MICRO CODE THAT THE KMC EXECUTES
965 ;*****
966 003644 000004          ST2:  SCOPE
967 003646 012737 000050 001160  MOV    #50,$TIMES          ;DO 50 ITERATIONS
968 .SBTTL KMC-11 MICRO-INITIALIZATION.
969 003654 004537 037534   JSR    RS,$LOAD           ;LOAD KMC-DIAG. MICRO-CODE
970 .GLOBL DRLPX1
971 003660 000000G        .WORD  DRLPX1
972 003662 012777 040000 175576  MOV    #BIT14,$KMA00      ;INITAILIZE LPA
973 003670 005000        CLR    RO
974 003672 005200        10$:  INC    RO                ;SHORT DELAY.
975 003674 001376        BNE    10$
976
977 003676 112777 000210 175564  MOVB   #BIT7!BIT3,$KMA01  ;SET RUN BIT ON KMC.
978
979 .SBTTL KMC-DIAG. SUB-TEST#1 CRAM WRITE ONES TEST
980 .REM  %
981 ;TEST #1
982 ;
983 ;
984 ;
985 ;
986 ;
987 ;
988 ;
989 ;
990 ;
991 ;
992 ;
993 ;
994 ;
995 ;
996 ;
997 ;
998 ;
999 003704 005037 001126          TST1: MOVE   #377,BREG        ;GET ALL ONES
1000 003710 012737 000377 001124  MOVE   BREG,OUT1 <0>     ;PUT ALL ONES INTO CRAM
1001 003716 117737 175544 001126  MOVE   BREG,OUT1 <2>
1002 003724 023737 001124 001126  MOVE   BREG,OUT1 <3>
1003 003732 001401          MOVE   BREG,OUT1 <4>
1004          MOVE   BREG,OUT1 <5>
1005          MOVE   BREG,OUT1 <6>
1006          MOVE   BREG,OUT1 <7>
1007          A1:  MOVE   INP1 <0> ,BREG  ;WHEN POP-11 DONE VERIFICATION,
1008          BZ     A1                ;IT WILL CLEAR CRAM ADDR#0
1009          %
1010          CLR    $BDDAT
1011          MOV    #377,$GDDAT        ;EXPECT 377 IN ADDR. #0
1012          MOVB   $KMA00,$BDDAT     ;READ ADDR 0
1013          CMP    $GDDAT,$BDDAT     ;GOOD READ?
1014          BEQ    11$              ;YES-NEXT SUB-TEST.
1015          ERROR  2                  ;CRAM WRITE ONES TEST
1016          11$: MOV    #177777,$GDDAT ;EXPECT 377,377, FROM ADDRS,2,3,4,5,6, AND 7
1017          MOV    $KMA02,$BDDAT     ;READ ADDRS 2 AND 3
1018          CMP    $BDDAT,$GDDAT     ;READ ALL ONES?
1019          BEQ    12$              ;YES-CHECK NEXT REG.
1020          ERROR  2                  ;CRAM WRITE ONES TEST, ADDR. 2 OR 3
1021          12$: MOV    $KMA04,$BDDAT ;READ ADDRS. 4 AND 5.
1022          CMP    $BDDAT,$GDDAT     ;READ ALL ONES?
1023          BEQ    13$              ;YES-NEXT REGS.
1024          13$:

```

```

1017
1018 004002 104002          ERROR 2          ;CRAM WRITE ONES TEST, ADDR. 4 OR 5
1019
1020 004004 017737 175472 001126 13$: MOV  @KMAD6,$BDDAT ;READ ADDRS. 6 AND 7.
1021 004012 023737 001126 001124  CMP  $BDDAT,$GDDAT ;READ ALL ONES?
1022 004020 001401          BEQ  14$          ;YES-NEXT TEST
1023
1024 004022 104002          ERROR 2          ;CRAM WRITE ONES TEST ADDR 6 OR 7.
1025
1026 004024 105077 175436 14$: CLR  @KMADO          ;TELL THE MICRO-PROGRAM TO
1027                                ;DO TEST #2.
1028
1029
1030          .SBTTL  KMC-DIAG SUB-TEST #2 CRAM WRITE ZEROES TEST
1031          .REM
1032          %
1033          TEST #2
1034          CRAM ZEROES WRITE TEST
1035          ENTERED WHEN PDP-11 ZEROES CRAM ADDR 0
1036
1037
1038          TST2: MOVE  #0,BREG          ;GET ZERO.
1039                MOVE  BREG,OUT1 <0> ;PUT INTO ALL CRAM.
1040                MOVE  BREG,OUT1 <2>
1041                MOVE  BREG,OUT1 <3>
1042                MOVE  BREG,OUT1 <4>
1043                MOVE  BREG,OUT1 <5>
1044                MOVE  BREG,OUT1 <6>
1045                MOVE  BREG,OUT1 <7>
1046
1047          A2:  MOVE  INP1 <0>,BREG    ;GET ZERO
1048                BZ   TST3          ;WHEN =377,PDP-11 READY FOR NEXT TEST.
1049                BR   A2
1050
1051          %
1052
1053 004030 000240          NOP
1054 004032 000240          NOP
1055 004034 005037 001124  CLR  $GDDAT          ;EXPECT ZERO REGISTERS.
1056 004040 117737 175422 001126  MOV  @KMADO,$BDDAT ;READ ADR.0
1057 004046 001404          BEQ  21$          ;IF ZERO, GOOD.
1058 004050 042737 177400 001126  BIC  #177400,$BDDAT
1059
1060 004056 104002          ERROR 2          ;CRAM WRITE ZERO TEST, ADDR.0
1061
1062 004060 017737 175406 001126 21$: MOV  @KMAD2,$BDDAT ;READ ADDRS. 2,3
1063 004066 001401          BEQ  22$          ;GOOD IF=0
1064
1065 004070 104002          ERROR 2          ;CRAM WRITE ZERO TEST, ADDR 2 OR 3
1066
1067 004072 017737 175400 001126 22$: MOV  @KMAD4,$BDDAT ;READ ADDRS 4,5
1068 004100 001401          BEQ  23$          ;GOOD IF=0
1069
1070 004102 104002          ERROR 2          ;CRAM WRITE ZERO TEST, ADDR. 4 OR 5
    
```

```

1071
1072 004104 017737 175372 001126 23$: MOV   @KMA06,$BDDAT ;READ ADDRS. 6,7
1073 004112 001401                BEQ   24$           ;GOOD IF=0
1074
1075 004114 104002                ERROR 2             ;CRAM WRITE ZERO TEST ADDR. 6,7.
1076
1077 004116 112777 000377 175342 24$: MOVB  #377, @KMADO ;TELL MICRO-CODE WE'RE READY FOR NEXT TEST.
1078
1079                .SBTTL  KMC-DIAG SUB TEST #3 BRANCH TEST
1080                .REM   %
1081
1082                | TEST #3
1083                | BRANCH TEST
1084                | ENTERED WHEN PDP-11 RETURNS 377 TO ADDR.0
1085                | NOTE WE HAVE TO ASSUME BR AND BZ WORK
1086                | OR WE COULDN'T HAVE GOT THIS FAR.
1087                | UPON SUCCESSFULL COMPLETION OF THIS TEST WE
1088                | PUT CODE 377 INTO CRAM ADDR 2
1089                | AND 0 INTO CRAM ADDR.0. ON FAILURE
1090                | WE ONLY ZERO CRAM ADDR.0.
1091
1092
1093
1094                TST3:  MOVE   # 376,BREG ;GET ZERO BIT0
1095                BB0    T3E           ;ERROR IF BR
1096                MOVE   # 375,BREG ;BIT 1=0
1097                BB1    T3E           ;ERROR IF BR
1098                MOVE   # 357,BREG ;BIT 4=0
1099                BB4    T3E           ;ERROR IF BR
1100                MOVE   # 177,BREG ;BIT 7=0
1101                BB7    T3E           ;ERROR IF BR
1102
1103                MOVE   # 1,BREG ;POSITIVE BR TEST
1104                BB0    A3            ;SHOULD BRANCH
1105                BR     T3E           ;IF NOT, ERROR.
1106                A3:   MOVE   # 2,BREG ;BR OR BIT 1
1107                BB1    B3            ;SHOULD BR.
1108                BR     T3E           ;IF NOT ERROR.
1109                B3:   MOVE   # 20,BREG ;BR ON BIT 4
1110                BB4    C3            ;SHOULD BR.
1111                BR     T3E           ;IF NOT ERROR.
1112                C3:   MOVE   # 200,BREG ;BR OF BIT 7
1113                BB7    D3            ;SHOULD BR.
1114                BR     T3E           ;IF NOT ERROR.
1115                D3:   MOVE   # 0,BREG ;TEST BZ NEGATIVE
1116                BZ    T3E           ;ERROR IF BR.
1117                MOVE   # 377,BREG ;POS BR
1118                BZ    E3            ;SHOULD BR.
1119                BR     T3E           ;ERROR IF NOT.
1120
1121                E3:   MOVE   BREG,OUT1 <2> ;PUT 377 IN OUTPUT REG #2
1122                BR     F3            ;IF BR INSTR FAILS TO WORK
1123                MOVE   # 0,BREG ;THEN A ZERO GETS PUT IN #2
1124                MOVE   BREG,OUT1 <2> ;A SIGN OF AN ERROR
    
```

```

1125
1126
1127
1128
1129
1130 004124 005000
1131 004126 105777 175334
1132 004132 001404
1133 004134 005200
1134 004136 001373
1135
1136 004140 104003
1137 004142 000550
1138
1139 004144 105777 175322
1140 004150 001002
1141
1142 004152 104003
1143 004154 000543
1144
1145 .SBTTL KMC-DIAG SUB-TEST #4 ALU TEST
1146 .REM %
1147 .TEST #4
1148
1149 ALU TEST
1150
1151 ENTERED WHEN PDP-11 PUTS A CODE 377 INTO
1152 CRAM ADDR 0
1153 ADDR 2=0 IF BAD
1154
1155 TST4: MOVE INP1 <0>,BREG ;GET CRAM ADDR0
1156 BZ A4 ;WHEN =377 DO TEST
1157 BR TST4
1158
1159 A4: MOV # 0,BREG ;GET=0
1160 MOV BREG,OUT1 <2>
1161 MOVE BREG,SPAD <4>
1162 DEC SPAD <4> ;MAKE = 377
1163 BZ B4 ;BR IF GOOD SEC.
1164 BR T4E
1165 B4: DEC SPAD <4> ;SEC = 376
1166 INC SPAD <4> ;INC = 377
1167 BZ C4 ;BR IF = 377
1168 BR T4E
1169 C4: MOVE # 377,BREG
1170 MOVE BREG,OUT1 <2> ;RETURN GOOD CODE
1171
1172
1173 T4E: MOVE # 0,BREG ;RETURN CODE 0
1174 MOVE BREG,OUT1 <0>
1175 %
1176
1177 004156 112777 000377 175302 40$: MOVB #377,AKMADD ;START TEST.
1178 004164 005000 CLR RO ;TIME OUT COUNTER

```

```

1179 004166 105777 175274      41$: TSTB@KMA00      ;DONE?
1180 004172 001404      BEQ      42$      ;YES-CHECK RESULTS
1181 004174 005200      INC      RD       ;NO-TIME OUT?
1182 004176 001373      BNE      41$      ;NO-LOOP.
1183
1184 004200 104003      ERROR    3        ;KMC-11 FAILED TO FINISH ALU TEST.
1185 004202 000530      BR      TSTS      ;
1186
1187 004204 122777 000377 175260  42$: CMPB    #377,@KMA02 ;DID IT DO IT RIGHT?
1188 004212 001401      BEQ      50$
1189
1190 004214 104003      ERROR    3        ;KMC-11 ALU TEST FAILED.
1191
1192      .SBTTL  KMC-DIAG SUB-TEST #5 NPR TEST ZERO TRANSFER
1193
1194      .REM    %
1195
1196      ;
1197      ;TEST 5
1198
1199      ;
1200      ;NPR TEST
1201      ;
1202      ;ENTERED IN LINE WHEN PDP-11 PUTS CODE 377 INTO
1203      ;CRAM ADDR.0
1204      ;DOES AN INPUT DATA XFERR FROM LOC 1124
1205      ;IN PDP-11 MEM TO LOC 1126
1206      ;DOES NO DATA CHECKING RETURNS
1207      ;CODE 0 TO CRAM ADDR.0 WHEN DONE.
1208
1209      TSTS:  MOVE   INPI <0>,BREG ;WAIT FOR PDP-11
1210           BZ     AS
1211           BR     TSTS
1212
1213      AS:   MOVE   # 2,BREG ;SET TO DO NPR IN
1214           MOVE   BREG,OUT0 <5> ;SET HIGH ADDR.
1215           MOVE   # 124,BREG ;GET LOW ADDR OF ADDR. 1124
1216           MOVE   BREG,OUT0 <4> ;SET LOW ADDR.
1217           MOVE   # 1,BREG ;SET TO NPR IN
1218           MOVE   BREG,OUT1 <10> ;CAUSE NPR IN
1219      BS:   MOVE   INPI <10>,BREG ;WAIT FOR NPR DONE.
1220           BBO
1221
1222           MOVE   INPD <0>,BREG ;GET LOW BYTE DATA.
1223           MOVE   BREG,OUT0 <2> ;OUTPUT NPR DATA REG.
1224           MOVE   INPD <1>,BREG ;GET HIGH BYTE.
1225           MOVE   BREG,OUT0 <3> ;SAVE FOR OUTPUT.
1226           MOVE   # 2,BREG ;GET OUTPUT ADDR.=1126
1227           MOVE   BREG,OUT0 <7>
1228           MOVE   # 126,BREG ;LOW PART.
1229           MOVE   BREG,OUT0 <6>
1230           MOVE   INPI <11>,SPAD <4> ;/--STN-
1231           AND    SPAD <4>,BREG,SPAD
1232           MOVE   # 0,BREG

```

```

1233 OR SPAD <4>,BREG,BREG
1234 MOVE BREG,OUT1 <11>
1235 MOVE # 21,BREG
1236 MOVE BREG,OUT1 <10> ;SET NPR OUT.
1237
1238 CS; MOVE INP1 <10>,BREG ;WAIT TILL DONE
1239 BBO CS
1240
1241 MOVE # 0,BREG ;TELL PDP-11 WE'RE DONE.
1242 MOVE BREG,OUT1 <0>
1243
1244 %
1245
1246 004216 005037 001124 50$: CLR $GDDAT ;CLEAR $GDDAT (LOC 1124)
1247 004222 012737 177777 001126 MOV #-1,$BDDAT
1248 004230 112777 000377 175230 MOVB #377,$KMA00 ;TELL MICRO-CODE TO GO!
1249 004236 005000 CLR RO ;CLEAR TIME OUT COUNTER.
1250
1251 004240 105777 175222 51$: TSTB $KMA00 ;KMC-MICROCODE DONE?
1252 004244 001403 BEQ 52$
1253 004246 005200 INC RO ;NO-TIME OUT OCCUR?
1254 004250 001373 BNE 51$
1255
1256 004252 104003 ERROR 3 ;KMC-11 FAILED TO FINISH NPR TEST.
1257
1258 004254 023737 001124 001126 52$: CMP $GDDAT,$BDDAT ;NPR ALL ZEROS CORRECTLY?
1259 004262 001401 BEQ 60$ ;YES-NEXT TEST.
1260
1261 004264 104002 ERROR 2 ;KMC-11 NPR TEST ZERO TRANSFER
1262
1263 .SBTTL KMC-DIAG SUB-TEST #6 NPR TEST ONES TRANSFER.
1264 .REM %
1265
1266 :TEST 6
1267
1268 : NPR TEST
1269
1270 : ENTERED INLINE WHEN PDP-11 PUTS CODE 377 INTO
1271 : CRAM ADDR.0
1272 : DOES AN INPUT DATA XFERR FROM LOC 1124
1273 : IN PDP-11 MEM TO LOC 1126
1274 : DOES NO DATA CHECKING RETURNS
1275 : CODE 0 TO CRAM ADDR.0 WHEN DONE.
1276
1277
1278 TST6: MOVE INP1 <0>,BREG ;WAIT FOR PDP-11
1279 BZ A6
1280 BR TST6
1281
1282 A6: MOVE # 2,BREG ;SET TO DO NPR IN
1283 MOVE BREG,OUT0 <5> ;SET HIGH ADDR.
1284 MOVE # 124,BREG ;GET LOW ADDR OF ADDR. 1124
1285 MOVE BREG,OUT0 <4> ;SET LOW ADDR.
1286 MOVE # 1,BREG ;SET TO NPR IN

```



```

1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315 004266 005037 001126
1316 004272 012737 177777 001124
1317
1318 004300 112777 000377 175160
1319
1320 004306 005000
1321
1322 004310 105777 175152
1323 004314 001404
1324 004316 005200
1325 004320 001373
1326
1327 004322 104003
1328 004324 000457
1329
1330 004326 023737 001124 001126
1331 004334 001401
1332
1333 004336 104002
1334
1335
1336
1337
1338
1339
1340

      MOVE BREG,OUT1 <10> ;CAUSE NPR IN
      MOVE INP1 <10>,BREG ;WAIT FOR NPR DONE.
      BBO B6

      MOVE INPO <0>,BREG ;GET LOW BYTE DATA.
      MOVE BREG,OUT0 <2> ;OUTPUT NPR DATA REG.
      MOVE INPG <1>,BREG ;GET HIGH BYTE.
      MOVE BREG,OUT0 <3> ;SAVE FOR OUTPUT.
      MOVE # 2,BREG ;GET OUTPUT ADDR.=1126
      MOVE BREG,OUT0 <7> ;LOW PART.
      MOVE # 126,BREG
      MOVE BREG,OUT0 <6>
      MOVE INP1 <11>,SPAD <4> ;/-STN-
      AND SPAD <4>,BREG,SPAD
      MOVE # 0,BREG
      OR SPAD <4>,BREG,BREG
      MOVE BREG,OUT1 <11>
      MOVE # 21,BREG
      MOVE BREG,OUT1 <10> ;SET NPR OUT.

      C6: MOVE INP1 <10>,BREG ;WAIT TILL DONE
           BBO C6

           MOVE # 0,BREG ;TELL PDP-11 WE'RE DONE
           MOVE BREG,OUT1 <0>

           %

      60$: CLR $BDDAT ;NOW XFERR $GDDAT TO $BDDAT
           MOV #177777,$GDDAT

           MOVB #377,$KMA00 ;START NPR XFERR

           CLR RO ;TIME OUT COUNTER

      61$: TSTB $KMA00 ;DONE NPR?
           BEQ 62$
           INC RO ;NO-TIME OUT?
           BNE 61$

           ERROR 3 ;KMC-11 NPR TEST FAILED TO FINISH.
           BR TST3 ;

      62$: CMP $GDDAT,$BDDAT ;ONES XFERR OK?
           BEQ 70$ ;YES-NEXT TEST.

           ERROR 2 ;KMC-11 NPR ONES XFERR FAILED.

      .SBTTL KMC-DIAG SUB-TEST #7 INTERRUPT TEST.
      .REM %

      ;TEST #7
      ;
      ; INTERRUPT TEST

```

1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394

```

:
: THIS TEST GENERATES TWO INTERRUPTS
:
: THE 1ST INTERRUPT TO VECTOR XXL WHEN
: CRAM ADDR.0 =377 BY PDP-11 CONTROL.
: THE 2ND INTERRUPT TO VECTOR XX4 WHEN
: CRAM ADDR0 =377 (1ST MODE=0 AGAIN)
:
TST7:  MOVE  INP1 <0>,BREG  ;WAIT TILL PDP-11 READY!
      BZ    A7
      BR    TST7

A7:    MOVE  INP1 <11>,SPAD <4>
      AND   SPAD <4>,BREG,SPAD
      MOVE  # 200,BREG
      OR    SPAD <4>,BREG,BREG
      MOVE  BREG,OUT1 <11> ;SET INTR TO ADDR. XX0
B7:    MOVE  INP1 <11>,BREG  ;WAIT TILL DONE
      BR    B7

      MOVE  # 0,BREG          ;TELL PDP-11 WE INTERRUPTED
      MOVE  BREG,OUT1 <0>    ;IN CASE WE DIDN'T.

C7:    MOVE  INP1 <0>,BREG  ;NOW WAIT FOR PDP-11 TO TELL
      BZ    D7              ;US TO INTR. TO VECTOR XX4
      BR    C7

D7:    MOVE  INP1 <11>,SPAD <4>
      AND   SPAD <4>,BREG,SPAD
      MOVE  # 300,BREG
      OR    SPAD <4>,BREG,BREG
      MOVE  BREG,OUT1 <11> ;SET INTR TO ADDR. XX4

E7:    MOVE  INP1 <11>,BREG  ;WAIT TILL DONE.
      BR    E7

      MOVE  # 0,BREG          ;TELL PDP-11 WE THOUGHT
      MOVE  BREG,OUT1 <0>    ;WE HAD INTERRUPTED!

      BR    .
      %

70$:   MOV   #72$,@VECT1     ;SET-UP FOR 1ST INTERRUPT.
      CLR   PS              ;LOWER PROCESSOR STATUS.
      MOVB  #377,@KMADD     ;TELL MICRO-CODE WE'RE READY!
      CLR   R0
      INCB  R0              ;TIME-OUT COUNT IN CASE OF NO INTERRUPT.
71$:   BPL  71$

      ERROR 3                ;KMC-FAILED TO INTERRUPT AT XX0
      BR    TST3            ;;

72$:   ADD  #4,SP           ;RESTORE STACK.
    
```

```

004340 012777 004372 175210
004346 005037 177776
004352 112777 000377 175106
004360 005000
004362 105200
004364 100376

004366 104003
004370 000435

004372 062706 U00004
    
```

1395	004376	012777	004444	175154		MOV	#74\$, @VECT2	; SET NEW INTERRUPT VECTOR
1396	004404	013777	001556	175144		MOV	VECT1, @VECT1	; RESTORE OLD VECTOR.
1397	004412	062777	000002	175136		ADD	#2, @VECT1	
1398	004420	005037	177776			CLR	PS	; ALLOW INTERRUPTS
1399	004424	112777	000377	175034		MOVB	#377, @KMA00	; TELL MICRO CODE WE'RE READY AGAIN.
1400	004432	005000				CLR	RO	
1401								
1402	004434	105200			73\$:	INCB	RO	; TIME OUT COUNTER IN CASE OF NO INTR
1403	004436	100376				BPL	73\$	
1404								
1405	004440	104003				ERROR	3	; KMC-FAILED TO INTERRUPT AT 004
1406	004442	000410				BR	TST3	::
1407								
1408	004444	062706	000004		74\$:	ADD	#4, SP	; RESTORE STACK
1409	004450	013777	001560	175102		MOV	VECT2, @VECT2	; RESTORE VECTOR.

H05

MAINDEC -11- DRLPA-A MACY11 27(654) 13-DEC-77 12:58 PAGE 29
DRLPA.P11 KMC-DIAG SUB-TEST #7 INTERRRUPT TEST.

SEQ 0059

1410 004456 062777 000002 175074 ADD #2,3VECT2
1411

```

1412
1413      ;:*****
1414      ;*TEST 3          *TEST SLAVE MICRO PROCESSOR START
1415
1416      ;*
1417      ;*IN THIS TEST WE'LL LOAD IN MICROCODE INTO THE KMC-11
1418      ;*THAT WILL ALLOW US TO TALK TO THE SLAVE MICRO-PROCESSOR.
1419
1420      ;*
1421
1422      ;*WHEN THE SLAVE MICRO-PROCESSOR IS INTIALIZED (IT GETS INITED
1423      ;*WHEN BIT 14 OF THE KMC'S CSR GET SET) IT SENDS US A VERSION
1424      ;*NUMBER THROUGH THE IPBM'S FAST PATH REGISTER IF THE
1425      ;*INITIAL CSR OF IPBM (AS SEEN BY THE SLAVE) IS CORRECT.
1426      ;*IF BAD, IT SEND CODE 377 THROUGH BOTH FAST PATH AND
1427      ;*SILO TO INDICATE AN ERROR.
1428
1429      ;:*****
1430      †ST3:  SCOPE
1431      004464 000004      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
1432      004466 012737 000010 001160
1433      004474 004537 037534      JSR      R5,$LOAD      ;LOAD MICRO CODE INTO KMC-11
1434      .GLOBL DRLPX0
1435      004500 000000G      .WORD   DRLPX0      ;FILE "DRLPX0"
1436
1437      004502 052777 040000 174756      BIS      #BIT14,$KMADD ;SET INIT BIT, WHEN THIS BIT

```

J05

MAINDEC -11- DRLPA-A
DRLPA.P11 T3

MACY11 27(654) 13-DEC-77 12:58 PAGE 31
*TEST SLAVE MICRO PROCESSOR START

SEQ 0061

1438
1439 004510 005000

CLR RO

:CLEARS IN SLAVE, RUN STARTS.

```

1440 004512 005200          1$: INC      RO          ;LITTLE DELAY LOOP.
1441 004514 001376          BNE     1$
1442
1443 004516 012777 104000 174742  MOV     #BIT15!BIT11, @KMADO ;CLEAR INTIT, SET RUN
1444 004524 005000          CLR     RO

```

L05

MAINDEC -11- ORLPA-A
DRLPA.P11 T3

MACY11 27(654) 13-DEC-77 12:58 PAGE 33
*TEST SLAVE MICRO PROCESSOR START

SEQ 0063

1445 004526 005200

2S: INC RD

;LITTLE DELAY.


```

1446 004530 001376          BNE      2$
1447
1448 004532 032777 000040 174726  BIT      #BITS, @KMA00 ;DID SLAVE SEND DATA THROUGH F.P.?
1449 004540 001410          BEQ      4$ ;BR IF YES (LOOKING AT IPBM CSR BIT 5)
1450 ;SENDS EITHER VERSION OR CODE 377.
1451
1452 004542 032777 000002 174716  BIT      #BIT-1, @KMA00 ;RIGHT NOW WE KNOW AN ERROR
1453 004550 001002          BNE      3$ ;HAS OCCURRED. IF THERE IS DATA
1454 ;IN THE SILO, MASTER'S IPBM
1455 ;CSR BIT 1=0
1456
1457 ;IPBM INIT ERROR
1458 004552 104004          ERROR    4 ;ERROR (IPBM) M8254 FAILED TO
1459 ;INITIALIZE PROPERLY. AS SEEN FROM
1460 004554 000427          BR      TST4 ;
1461
1462
1463 004556 104004          3$: ERROR    4 ;ERROR IPBM OR (MOST LIKELY)
1464 ;SLAVE MICRO PROCESSOR FAILED
1465 ;TO START PROPERLY.
1466 004560 000425          BR      TST4 ;
1467
1468 004562 112777 000004 174702  4$: MOVB    #4, @KMA02 ;ISSUE CMMD TO READ FAST PATH
1469
1470 004570 122777 000377 174674  5$: CMPB    #377, @KMA02 ;WAIT FOR READ.
1471 004576 001374          BNE      5$

```


MAINDEC -11- DRLPA-A
DRLPA.P11 T3

MACY11 27(654) 13-DEC-77 12:58 PAGE 36
*TEST SLAVE MICRO PROCESSOR START

SEQ 0066

1479 004620 000405

BR TST4

::

```

1480
1481 004622 123737 001124 001126 6$: CMPB $GDDAT,$BDDAT ; DOES THE MICRO-CODE VERSION
1482 ; NUMBER AGREE WITH THE
1483 ; VERSION WE EXPECT?
1484 004630 001401 BEQ TST4 ;
1485 ;
1486 004632 104005 ERROR 5 ; WRONG MICRO-CODE VERSION #
1487 ; RETURNED BY SLAVE MICRO-PROCESSOR.
1488 ; NOTE: (1) YOU COULD BE RUNNING THE
1489 ; WRONG VERSION OF THE DIAGNOSTIC
1490 ; (2) SLAVE MICRO COULD BE BAD
1491 ; OR (3) M8254 MIGHT BE BAD
1492 ; (FAST PATH)
1493
1494 .SBTTL **PHASE 2 IPBM/DMC TESTING.

```

1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548

*TEST 4 *TEST DATA LOOP BACK THROUGH IPBM,MB254 MODULE
*
*IN THIS TEST WE WILL SEND DATA THROUGH THE SILO
*AND FAST PATH. TO UNDERSTAND HOW WE'RE ABLE TO DO THIS,
*YOU MUST FIRST UNDERSTAND WHAT THE SLAVE MICRO-PROCESSOR
*IS DOING.
* WHEN THE SLAVE RECEIVES CODE 200 IN ITS FAST PATH REGISTER
* WHILE IT IS COMMAND MODE, IT MONITORS THE F.P. AND READS
* THE F.P. INPUT (WHEN DATA IS PRESENT) AND LOOPS IT BACK TO ITS
* F.P. OUTPUT. IT DOES THIS 256 (DECIMAL) TIMES. NEXT THE SLAVE
* MONITORS THE SILO REGISTER AND LOOPS SILO DATA BACK
* FOR 256 TIMES. AT THE CONCLUSION OF THIS, IT RETURNS TO
* COMMAND MODE.

*THIS TEST USES KMC-11 MICRO-CODE FILE "DRLPX0" PRE LOADED
*INTO THE KMC-11 BY THE PREVIOUS TEST.

*THIS MICRO-CODE RUNS CONTINUOUS READING THE IPBM STATUS
*PUTTING IT IN KMC-11 ADDR.0, AND LOOKING FOR A COMMAND TO BE
*PUT INTO KMC-11 ADDR.2. IT RETURNS CODE
*"377" TO ADDR.2 WHEN FINISHED COMMAND. DATA IS XFERRED
*THROUGH KMC-11 ADDR.4.

*HERE IS A LIST OF COMMANDS FOR FILE "DRLPX0"
*IF YOU HALT AT THE END OF THIS TEST YOU MAY USE
*THIS MICRO-CODE. CAUTION: BITIS OF KMC-11 ADDR. MUST REMAIN
*SET WHEN A COMMAND IS GIVEN (THIS IS THE "RUN" BIT) (INITIALIZE
*ALSO CLEARS BIT). I USE BYTE ADDRESSING TO AVOID CLEARING IT.

"DRLPX0" COMMAND	FUNCTION
1	NOP
2	READ SILO, PUT IN LOW BYTE KMC ADDR.4
3	WRITE SILO, DATA IN LOW BYTE OF KMC ADDR.4
4	READ FAST PATH REG., PUT IN LOW BYTE OF KMC ADDR
5	WRITE FAST PATH REG., DATA IN LOW BYTE OF KMC AD

*COMMAND "200" SENT TO THE SLAVE THROUGH THE FAST PATH
*REG TO PUT IT INTO LOOP BACK MODE.

*YOU MAY NOT LOOP ON A SPECIFIC DATA PATTERN FAILURE
*YOU MAY USE LOOP ON TEST OPTION (SWR BIT14).
*IF THIS TEST FAILS YOU SHOULD RUN THE MB254(IPBM) DIAGNOSTIC.

```
*****
*ST4:  SCOPE
      MOV    #10,$TIMES      ;;DO 10 ITERATIONS
      CLR    20$            ;DATA LOOP COUNTER
      MOV    #1,$SLPADR     ;SET FOR LOOP ON ADDR.
      MOV    #1,$SLPADR     ;IN CASE LOOP ERROR IS USED.
      MOV    #0,21$         ;SET INITIAL DATA PATTERN.
      1$:  TST    20$        ;CHECK FOR LEGAL TEST ENTRY.
      BNE   2$             ;BAD ENTRY, PROBABLY FROM "LOOP ON ERROR"
```

004634 000004
004636 012737 000010 001160
004644 005037 005266
004650 012737 000001 001106
004656 012737 004672 001106
004664 012737 000000 005270
004672 005737 005266
004676 001006

```

1549 ;WE'LL TRY TO rECOVER
1550
1551 004700 112777 000200 174570      MOVB  #200,2KMA04 ;PUT "LOOP BACK" CMMO INTO XFERR REG.
1552 004706 112777 000005 174556      MOVB  #5,2KMA02 ;MAKE "DRLPX0" GIVE FAST PATH CMMO TO
1553 ;SLAVE.
1554 ;AT THIS POINT, SLAVE WILL
1555 ;LOOP 256 (DECIMAL) PIECES OF DATA
1556 ;THROUGH THE FAST PATH REG.
1557
1558 004714 105737 005267      2$: TSTB 20$+1 ;IF IN SILO SECTION, EXIT F.P. SECTION.
1559 004720 001064      BNE 10$
1560 004722 004737 034062      JSR PC,WIR ;WAIT FOR PROGRAM "DRLPX0" TO BECOME
1561 ;READY FOR ANOTHER COMMAND.
1562
1563 004726 012737 000052 001124      MOV #52,$GDDAT ;AT THIS POINT, EXPECT IPBM (MASTER
1564 004734 005037 001126      CLR $BDDAT ;SIDE) CSR=52
1565 004740 117737 174522 001126      MOVB 2KMA00,$BDDAT ;READ CSR ("DRLPX0" REFLECTS CSR IN
1566 ;KMC'S ADDR.0)
1567 004746 123737 001124 001126      CMPB $GDDAT,$BDDAT ;-RIGHT CODE?
1568 004754 001401      BEQ 3$ ;YES-SKIP ERROR.
1569
1570 004756 104005      ERROR 5 ;CSR BAD. IPBM MASTER CSR S/B 52.
1571
1572 004760 113777 005270 174510      3$: MOVB 21$,2KMA04 ;PUT DATA PATTERN IN PRG. "DRLPX0"
1573 004766 005237 005266      INC 20$ ;XFERR REG.
1574 004772 112777 000005 174472      MOVB #5,2KMA02 ;ISSUE WRITE F.P. CMMO.
1575 005000 004737 034062      JSR PC,WIR ;WAITE FOR "DRLPX0" TO BECOME READY AGAIN.
1576 ;AT THIS POINT IN TIMES THE

```

MAINDEC -11- DRLPA-A
DRLPA.P11 T4

MACY11 27(654) 13-DEC-77 12:58 PAGE 40
*TEST DATA LOOP BACK THROUGH IPBM, M8254 MODULE

SEQ 0070

```

1577
1578
1579 005004 005000
1580 005006 032777 000040 174452 4$: CLR RO
1581 005014 001403 BEQ #BITS, @KMADO
1582 005016 105200 INCB RO
1583 005020 100372 BPL 4$
1584
1585
1586 005022 104004 ERROR 4
1587
1588
1589
1590 005024 112777 000004 174440 5$: MOVB #4, @KMAD2
1591 005032 004737 034062 JSR PC, WIR
1592
1593
1594 005036 117737 174434 001126 MOVB @KMAD4, $BDDAT
1595 005044 113737 005270 001124 MOVB 21$, $GDDAT

```

```

; SLAVE SHOULD "SEE" DATA IN THE
; F.P. AND LOOP IT BACK TO US.
; WAIT TILL F.P. HAS DATA TO BE READ.
; WHEN READY, EXIT LOOP,
; DON'T WAIT TOO AWFUL LONG!
; IF INC RO OVERFLOW, THE SLAVE
; NEVER WILL RETURN DATA.
; SLAVE MICRO-PROCESSOR FAILED TO
; LOOP BACK DATA THROUGH FAST PATH REG.
; DATA IN F.P. READY TO BE READ, BETS
; TELL "DRLPX0" TO READ THE F.P..
; WAITE FOR DRLPX0.
; READ DATA
; 21$ CONTAINED DATA SENT

```

1596											
1597	005052	123737	001124	001126		CMPB	\$GDDAT,\$BDDAT				;DID DATA LOOP BACK OK?
1598	005060	001401				BEQ	6\$				
1599											
1600	005062	104007				ERROR	7				;FAST PATH DATA ERROR
1601											
1602	005064	105237	005270		6\$:	INCB	21\$;UPDATE LOOP BACK PATTERN.
1603											;NOTE: IF YOU REALY WANT TO, YOU
1604											;CAN MODIFY THIS SECTION OF TEST
1605											;TO SEND ONLY A SPECIFIC PATTERN.
1606											;TO DO THIS, MODIFY
1607											;THE INSTRUCTOR "MOV #0,21\$" TO
1608											;ANY STARTING PATTERN YOU WANT.
1609											;MODIFIED TO "NOP,NOP" IF YOU WISH
1610											;TO SEND THE SAME PATTERN, OR
1611											;TO ANY OTHER AUNRY INSTRUCTION THAT
1612											;WOULD MODIFY THE PATTERN.
1613											
1614	005070	000711				BR	2\$;IF NOT DONE 256 TIMES LOOP.
1615											;WARNING: THE PROGRAM CAN ONLY
1616											;SEND 256 DATA BYTES THROUGH
1617											;THE FAST PATH REG. BEFORE IT MUST
1618											;LOOP 256 DATA BYTES THROUGH
1619											;THE SILO.
1620											
1621	005072				10\$:						;WE COME HERE ONLY AFTER
1622											;256 DATA BYTES HAVE BEEN
1623											;XFERRED THROUGH THE FAST PATH
1624											;NOW WE'LL LOOP 256 OF DATA
1625											;THROUGH THE SILO.
1626	005072	012737	000000	005270		MOV	#0,21\$;PUT NEW PATTERN IN PATTERN LOC
1627											;IF YOU DESIRE ANOTHER STARTING
1628											;PATTERN, YOU MAY REPLACE THE 0
1629											;WITH IT (REMEMBER ONLY 8 BITS LONG).
1630											
1631	005100	032737	001000	005266	11\$:	BIT	#BIT9,20\$;DONE?
1632	005106	001064				BNE	18\$;YES-EXIT.
1633	005110	004737	034062			JSR	PC,WIR				;MAKE SURE "DRLPX0" IS READY.
1634											
1635	005114	012737	000052	001124		MOV	#52,\$GDDAT				;AT THIS POINT, EXPECT IPBM (MASTER
1636	005122	005037	001126			CLR	\$BDDAT				;SIDE) SCR=52
1637	005126	117737	174334	001126		MOVB	@KMADD,\$BDDAT				;READ CSR ("DRLPX0" REFLECTS CSR IN
1638											;KMC'S ADDR.0)
1639											
1640	005134	123737	001124	001126		CMPB	\$GDDAT,\$BDDAT				; -RIGHT CODE?
1641	005142	001401				BEQ	12\$; -YES-SKIP ERROR.
1642											
1643	005144	104005				ERROR	5				;CSR BAD, IPBM MASTER SIDE, S/B 52
1644											;WE WERE OK WHILE WORKING
1645											;WITH FAST PATH, CSR WENT BAD DURING
1646											;SILO TESTING.
1647	005146	113777	005270	174322	12\$:	MOVB	21\$,@KMAD4				;PUT DATA PATTERN IN PROG "DRLPX0"
1648	005154	005237	005266			INC	20\$;XFERR REG.
1649	005160	112777	000003	174304		MOVB	#3,@KMAD2				;MAKE "DRLPX0" WRITE SILO.


```

1650 005166 004737 034062          JSR      PC,WIR          ;WAIT TILL "DRLPX0" READY.
1651                                     ;NOW-WE PUT DATA IN SILO.
1652                                     ;SLAVE MICRO-CODE READS SILO
1653                                     ;DATA IN AND PUTS IT IN SILO
1654                                     ;OUT. THATS WHERE WE PICK IT UP.
1655
1656 005172 005000          CLR      RO
1657 005174 032777 000002 174264 13$:  BIT     #BIT1,@KMADD    ;WAIT TILL SILO HAS DATA TO BE READ
1658 005202 001403          BEQ     14$              ;WHEN READY, EXIT LOOP.
1659 005204 105200          INCB   RO                ;DON'T WAIT TOO LONG!
1660 005206 100372          BPL     13$              ;IF INC RO OVERFLOW, THE SLAVE
1661                                     ;NEVER WILL RETURN DATA.
1662
1663 005210 104004          ERROR   4                ;SLAVE MICRO FAILED TO LOOP BACK
1664                                     ;DATA THROUGH SILO..
1665
1666 005212 112777 000002 174252 14$:  MOVB   #2,@KMAD2        ;DATA IN SILO READY TO BE READ, TELL
1667                                     ;DRLPX0 TO READ IT.
1668 005220 004737 034062          JSR      PC,WIR          ;WAIT FOR DRLPX0 READY
1669
1670 005224 117737 174246 001126  MOVB   @KMAD4,$BDDAT    ;GET DATA.
1671 005232 113737 005270 001124  MOVB   21$,$GDDAT      ;GET S/B
1672
1673 005240 123737 001124 001126  CMPB   $GDDAT,$BDDAT   ;DID DATA LOOP BACK OK?
1674 005246 001401          BEQ     15$
1675
1676 005250 104006          ERROR   6                ;SILO DATA ERROR.
1677

```

```

1678
1679 005252 105237 005270      15$:  INCB  21$
1680
1681
1682
1683
1684
1685
1686
1687
1688 005256 000710           BR    11$
1689
1690
1691
1692
1693
1694
1695
1696
1697 005260 005037 005266      18$:  CLR  20$
1698 005264 000402           BR    TST5
1699

```

```

;UPDATE LOOP BACK PATTERN
;NOTE: IF YOU REALLY WANT TO,
;YOU CAN MODIFY THIS
;SECTION OF TEST TO SEND ONLY A
;SPECIFIC PATTERN.
;TO DO THIS, CHANGE THE
;INSTRUCTION "INCB 21$" (2 WORDS)
;ALONG WITH THE "MOV#0,21$"
;
;IF NOT DONE 256 TIMES, LOOP.
;WARNING: THIS PROGRAM MUST
;SEND 256 DATA LOOP BACK PATTERNS
;THROUGH THE SILO REG SO
;THAT THE SLAVE MICRO CODE
;WILL EXIT FROM LOOP BACK
;MODE TO COMMAND MODE.
;
;EXIT, ALL DONE!
;;

```

J06

MAINDEC -11- DRLPA-A
DRLPA.P11 T4

MACY11 27(654) 13-DEC-77 12:58 PAGE 44
*TEST DATA LOOP BACK THROUGH IPBM, M8254 MODULE

SEQ 0074

1700 005266 000000
1701 005270 000000

20\$: .WORD 0
21\$: .WORD 0

:CONTAINS CURRENT LOOP COUNT.
:CONTAINS CURRENT LOOP BACK PATTERN.

K06

MAINDEC -11- DRLPA-A
DRLPA.P11 T4

MACY11 27(654) 13-DEC-77 12:58 PAGE 45
*TEST DATA LOOP BACK THROUGH IPBM, MB254 MODULE

SEQ 0075

1702
1703

.SBTTL PHASE 3 I/B BUS TESTING.

1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758

*TEST 5 I/O DEVICE TEST-AD11-K OPTION

*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.

*
* IN THIS TEST, WE WILL EXERCISE THE AD11-K
* IF DVID1 IS UNALTERED, WE WILL DO THIS TEST SENCE THE
* AD11-K IS A DEFAULT I/O DEVICE. IF AN AD11-K IS NOT ON THE
* I/O BUS, YOU MUST DELETE IT FROM SR.

- * TESTS:
- * 1. MAKE SURE AD11-K RESPONDS TO ADDRESS.
* (NOTE IF THE AD11-K ON YOU I/O BUS HAS NON-STANDARD
* ADDRESS, PLEASE REFLECT THE NEW ADDRESS IN ADDRESS "STREG")
 - * 2. MAKE SURE WE CAN WRITE A ZERO INTO CSR (EXCEPT BIT 7:15).
 - * 3. WRITE BITS 13,11,9,6 MAKE SURE THEY WRITE.
 - * 4. WRITE BITS 12,10,8,4 MAKE SURE THEY WRITE.
 - * 5. SET A/D START, MAKE SURE IT CLEARS, DONE FLAG SETS.
 - * 6. SET A/D START AGAIN, MAKE SURE ERROR FLAG SETS,
 - * 7. MAKE SURE ERROR FLAG CAN BE CLEARED.

*ST5: SCOPE

005272 000004
005274 032737 000001 001562
005302 001002
005304 000137 005750
005310
005310 005037 001126

BIT #BIT0,SR1 ;IS THIS DEVICE SELECTED FOR TEST?
;NOTE: DEFAULT=YES.
BNE 10\$
JMP 7\$
10\$: CLR \$BDDAT
;* MOV \$BDDAT,\$STREG1 ;/ PUT DATA FROM \$BDDAT TO DEVICE REG STREG1
;OK-WHAT WE JUST DID IS ASKED THE
;SLAVE MICRO-CODE TO WRITE THE CONTENTS

```

1759                                     ;OF THE AD11K'S CSR TO ZEROES.
1760                                     ;IF THE AD11K FAILS TO RESPOND TO ITS
1761                                     ;ADDRESS-THE SLAVE WILL SEND US AN
1762                                     ;ERROR CODE THAT CAUSES US TO SET $AERR=1
1763                                     ;IN THE SUPPORT ROUTINES.
1764
1765 005324 005737 037532                 TST   $AERR   ;DEVICE PRESENT?
1766 005330 001403                         BEQ   11$    ;YES-CONTINUE
1767 005332 104010                         ERROR 10     ;A/D #1 FAILED TO RESPOND
1768 005334 000137 005750                 JMP   7$    ;TO ADDR.
1769 005340
1770
1771                                     11$:
1772                                     ;*
1772 005350 042737 100200 001126           MOV   @STREG1,$BDDAT ;/READ DEVICE REG STREG1,PUT DATA IN $BDDAT.
1773 005356 005037 001124                 BIC   #BIT15!BIT7,$BDDAT ;CLEAR OUT FLAGS
1774 005362 005737 001126                 CLR   $GDDAT
1775 005366 001402                         TST   $BDDAT ;IS CSR ZERO?
1776                                     BEQ   1$
1777 005370 104010                         ERROR 10     ;FAILED TO ZERO AD11K CSR #1
1778 005372 000572                         BR    TST6
1779 005374 012737 025100 001124         1$: MOV   #BIT13!BIT11!BIT9!BIT6,$GDDAT ;NOW TRY A NEW BIT PATTERN
1780
1781                                     ;*
1782                                     ;*
1782 005422 042737 100200 001126           MOV   $GDDAT,@STREG1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG1
1783 005430 023737 001124 001126           ;* MOV   @STREG1,$BDDAT ;/READ DEVICE REG STREG1,PUT DATA IN $BDDAT.
1784 005436 001402                         BIC   #BIT15!BIT7,$BDDAT ;IGNORE FLAGS
1785 005422 042737 100200 001126           CMP   $GDDAT,$BDDAT ;BIT SET OK?
1786 005430 023737 001124 001126           BEQ   2$
1787 005436 001402
1788
1789 005440 104010                         ERROR 10     ;FAILED TO WRITE #1 AD11K CSR CORRECTLY
1790 005442 000546                         BR    TST6
1791
1792 005444 012737 012420 001124         2$: MOV   #BIT12!BIT10!BIT8!BIT4,$GDDAT ;NOW TRY A NEW BIT PATTERN.
1793
1794                                     ;*
1795                                     ;*
1795 005472 042737 100200 001126           MOV   $GDDAT,@STREG1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG1
1796 005500 023737 001124 001126           ;* MOV   @STREG1,$BDDAT ;/READ DEVICE REG STREG1,PUT DATA IN $BDDAT.
1797 005506 001402                         BIC   #BIT15!BIT7,$BDDAT ;IGNORE FLAGS
1798 005472 042737 100200 001126           CMP   $GDDAT,$BDDAT ;BITS SET OK?
1799 005500 023737 001124 001126           BEQ   3$
1800 005506 001402
1801
1802 005510 104010                         ERROR 10     ;FAILED TO WRITE #1 AD11K CSR CORRECTLY.
1803 005512 000522                         BR    TST6
1804
1805 005514 005037 001124                 3$: CLR   $GDDAT ;NOW CLEAR ALL BITS SET
1806
1807                                     ;*
1808                                     ;*
1807 005540 042737 000200 001126           MOV   $GDDAT,@STREG1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG1
1808 005546 005737 001126                 ;* MOV   @STREG1,$BDDAT ;/READ DEVICE REG STREG1,PUT DATA IN $BDDAT.
1809 005552 001402                         BIC   #BIT7,$BDDAT ;IGNORE DONE FLAG.
1810 005540 042737 000200 001126           TST   $BDDAT ;CSR CLEAR?
1811 005546 005737 001126                 BEQ   4$
1812 005552 001402

```

```

1813
1814 005554 104010          ERROR 10          ;#1 AD11K CSR FAILED TO CLEAR.
1815 005556 000500          BR      TST6          ;;
1816
1817 005560                  4$:
1818
1819                      ;*      MOV      @ADBUF1,MYTEMP ;/READ DEVICE REG ADBUF1,PUT DATA IN MYTEMP.
1820 005570 012737 000001 001124      MOV      #BIT0,$GDDAT ;NOW WE'LL SET THE A/D START BIT.
1821
1822                      ;*      MOV      $GDDAT,@STREG1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG1
1823 005606 012737 000200 001124      MOV      #BIT7,$GDDAT ;WHEN WE READ CSR BACK EXPECT
1824                                     ;START BIT TO CLEAR, DONE TO SET.
1825
1826                      ;*      MOV      @STREG1,$BDDAT ;/READ DEVICE REG STREG1,PUT DATA IN $BDDAT.
1827
1828 005624 023737 001124 001126      CMP      $GDDAT,$BDDAT ;START OK?
1829 005632 001402          BEQ      5$
1830
1831 005634 104010          ERROR 10          ;CSR BAD AFTER #1 A/D START
1832                                     ;BIT 0 SHOULD CLEAR ONLY A/D DONE SET.
1833 005636 000450          BR      TST6          ;;
1834
1835 005640 012737 000001 001124      5$: MOV      #BIT0,$GDDAT ;OK-NOW WE'RE GONNA START ANOTHER
1836                                     ;A/D CONVERSION WITHOUT READING
1837                                     ;THE RESULTS OF THE LAST ONE. THIS
1838                                     ;SHOULD CAUSE THE ERROR FLAG
1839                                     ;AS WELL AS THE DONE FLAG TO SET.
1840
1841                      ;*      MOV      $GDDAT,@STREG1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG1
1842
1843
1844                      ;*      MOV      @STREG1,$BDDAT ;/READ DEVICE REG STREG1,PUT DATA IN $BDDAT.
1845 005666 012737 100200 001124      MOV      #BIT15!BIT7,$GDDAT ;S/B
1846
1847 005674 023737 001124 001126      CMP      $GDDAT,$BDDAT ;IS 15 AND 7 SET?
1848
1849 005702 001401          BEQ      6$
1850
1851 005704 104010          ERROR 10          ;ERROR #1 AD11K BIT 15 AND 7 SHOULD BE SET.
1852
1853
1854 005706 005037 001124          6$: CLR      $GDDAT          ;MAKE SURE ERROR FLAG CLEARS
1855
1856                      ;*      MOV      $GDDAT,@STREG1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG1
1857
1858                      ;*      MOV      @STREG1,$BDDAT ;/READ DEVICE REG STREG1,PUT DATA IN $BDDAT.
1859 005732 042737 000200 001126      BIC      #BIT7,$BDDAT ;IGNORE DONE FLAG.
1860 005740 005737 001126          TST      $BDDAT          ;IS CSR CLEAR?
1861 005744 001401          BEQ      7$
1862 005746 104010          ERROR 10          ;AD11K CSR FAILED TO CLEAR.
1863
1864 005750          7$:
1865
1866                      ;*      MOV      @ADBUF1,MYTEMP ;/READ DEVICE REG ADBUF1,PUT DATA IN MYTEMP.

```


1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920

*TEST 6 I/O DEVICE TEST-AD11-K OPTION

*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.

IN THIS TEST, WE WILL EXERCISE THE AD11-K
IF DVID1 IS UNALTERED, WE WILL DO THIS TEST SENCE THE
AD11-K IS A DEFAULT I/O DEVICE. IF AN AD11-K IS NOT ON THE
I/O BUS, YOU MUST DELETE IT FROM SR.

TESTS:

1. MAKE SURE AD11-K RESPONDS TO ADDRESS.
(NOTE IF THE AD11-K ON YOU I/O BUS HAS NON-STANDARD
ADDRESS, PLEASE REFLECT THE NEW ADDRESS IN ADDRESS "STREG")
2. MAKE SURE WE CAN WRITE A ZERO INTO CSR (EXCEPT BIT 7:15).
3. WRITE BITS 13,11,9,6 MAKE SURE THEY WRITE.
4. WRITE BITS 12,10,8,4 MAKE SURE THEY WRITE.
5. SET A/D START, MAKE SURE IT CLEARS, DONE FLAG SETS.
6. SET A/D START AGAIN, MAKE SURE ERROR FLAG SETS,
7. MAKE SURE ERROR FLAG CAN BE CLEARED.

*STB: SCOPE

005760	000004		
005762	032737	000020	001562
005770	001002		
005772	000137	006436	
005776			
005776	005037	001126	

```

BIT      #BIT4,SR1      ;IS THIS DEVICE SELECTED FOR TEST?
                        ;NOTE: DEFAULT=YES.
BNE      10$
JMP      7$
10$:
CLR      $BDDAT
;*      MOV      $BDDAT, $STREG2  ;/ PUT DATA FROM $BDDAT TO DEVICE REG STREG2
                        ;OK-WHAT WE JUST DID IS ASKED THE
                        ;SLAVE MICRO-CODE TO WRITE THE CONTENTS

```

; OF THE AD11K'S CSR TO ZEROES.
; IF THE AD11K FAILS TO RESPOND TO ITS
; ADDRESS-THE SLAVE WILL SEND US AN
; ERROR CODE THAT CAUSES US TO SET \$AERR=1
; IN THE SUPPORT ROUTINES.

; DEVICE PRESENT?
; YES-CONTINUE
; A/D #2 FAILED TO RESPOND
; TO ADDR.

; / READ DEVICE REG STREG2, PUT DATA IN \$BDDAT.

; CLEAR OUT FLAGS

; IS CSR ZERO?

; FAILED TO ZERO AD11K CSR #2

; NOW TRY A NEW BIT PATTERN

; / PUT DATA FROM \$GDDAT TO DEVICE REG STREG2

; / READ DEVICE REG STREG2, PUT DATA IN \$BDDAT.

; IGNORE FLAGS

; BIT SET OK?

; FAILED TO WRITE #2 AD11K CSR CORRECTLY

;;

; NOW TRY A NEW BIT PATTERN.

; / PUT DATA FROM \$GDDAT TO DEVICE REG STREG2

; / READ DEVICE REG STREG2, PUT DATA IN \$BDDAT.

; IGNORE FLAGS

; BITS SET OK?

; FAILED TO WRITE #2 AD11K CSR CORRECTLY.

;;

; NOW CLEAR ALL BITS SET

; / PUT DATA FROM \$GDDAT TO DEVICE REG STREG2

; / READ DEVICE REG STREG2, PUT DATA IN \$BDDAT.

; IGNORE DONE FLAG.

; CSR CLEAR?

1921
1922
1923
1924
1925
1926
1927 006012 005737 037532
1928 006016 001403
1929 006020 104010
1930 006022 000137 006436
1931 006026
1932
1933
1934 006036 042737 100200 001126
1935 006044 005037 001124
1936 006050 005737 001126
1937 006054 001402
1938
1939 006056 104010
1940 006060 000572
1941 006062 012737 025100 001124
1942
1943
1944
1945
1946
1947 006110 042737 100200 001126
1948 006116 023737 001124 001126
1949 006124 001402
1950
1951 006126 104010
1952 006130 000546
1953
1954 006132 012737 012420 001124
1955
1956
1957
1958
1959
1960 006160 042737 100200 001126
1961 006166 023737 001124 001126
1962 006174 001402
1963
1964 006176 104010
1965 006200 000522
1966
1967 006202 005037 001124
1968
1969
1970
1971
1972 006226 042737 000200 001126
1973 006234 005737 001126
1974 006240 001402

11\$:

;*
;*
;*
1\$:
;*
;*
2\$:
;*
;*
3\$:
;*
;*
4\$:

TST \$AERR
BEQ 11\$
ERROR 10
JMP 7\$

MOV \$STREG2,\$BDDAT
BIC #BIT15!BIT7,\$BDDAT
CLR \$GDDAT
TST \$BDDAT
BEQ 1\$

ERROR 10
BR TST7
MOV #BIT13!BIT11!BIT9!BIT6,\$GDDAT

MOV \$GDDAT,\$STREG2

MOV \$STREG2,\$BDDAT
BIC #BIT15!BIT7,\$BDDAT
CMP \$GDDAT,\$BDDAT
BEQ 2\$

ERROR 10
BR TST7

MOV #BIT12!BIT10!BIT8!BIT4,\$GDDAT

MOV \$GDDAT,\$STREG2

MOV \$STREG2,\$BDDAT
BIC #BIT15!BIT7,\$BDDAT
CMP \$GDDAT,\$BDDAT
BEQ 3\$

ERROR 10
BR TST7

CLR \$GDDAT

MOV \$GDDAT,\$STREG2

MOV \$STREG2,\$BDDAT
BIC #BIT7,\$BDDAT
TST \$BDDAT
BEQ 4\$

```

1975
1976 006242 104010          ERROR 10          ;#2 AD11K CSR FAILED TO CLEAR.
1977 006244 000500          BR      TST7      ;;
1978
1979 006246                4$:
1980
1981                ;*      MOV      @ADBUF2,MYTEMP ;/READ DEVICE REG ADBUF2,PUT DATA IN MYTEMP.
1982 006256 012737 000001 001124 ;*      MOV      #BIT0,$GDDAT ;NOW WE'LL SET THE A/D START BIT.
1983
1984                ;*      MOV      $GDDAT,@STREG2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG2
1985 006274 012737 000200 001124 ;*      MOV      #BIT7,$GDDAT ;WHEN WE READ CSR BACK EXPECT
1986                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
1987
1988                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
1989
1990 006312 023737 001124 001126      CMP      $GDDAT,$BDDAT ;START OK?
1991 006320 001402          BEQ      5$
1992
1993 006322 104010          ERROR 10          ;CSR BAD AFTER #2 A/D START
1994                BR      TST7      ;BIT 0 SHOULD CLEAR ONLY A/D DONE SET.
1995 006324 000450          BR      TST7      ;;
1996
1997 006326 012737 000001 001124      5$: MOV      #BIT0,$GDDAT ;OK-NOW WE'RE GONNA START ANOTHER
1998                ;*      MOV      $GDDAT,@STREG2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG2
1999                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2000                ;*      MOV      #BIT15!BIT7,$GDDAT ;S/B
2001                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2002                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2003                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2004                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2005                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2006 006354 012737 100200 001124 ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2007                ;*      MOV      #BIT15!BIT7,$GDDAT ;S/B
2008
2009 006362 023737 001124 001126      CMP      $GDDAT,$BDDAT ;IS 15 AND 7 SET?
2010
2011 006370 001401          BEQ      6$
2012
2013 006372 104010          ERROR 10          ;ERROR #2 AD11K BIT 15 AND 7 SHOULD BE SET.
2014
2015
2016 006374 005037 001124          6$: CLR      $GDDAT ;MAKE SURE ERROR FLAG CLEARS
2017
2018                ;*      MOV      $GDDAT,@STREG2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG2
2019                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2020                ;*      MOV      @STREG2,$BDDAT ;/READ DEVICE REG STREG2,PUT DATA IN $BDDAT.
2021 006420 042737 000200 001126 ;*      BIC      #BIT7,$BDDAT ;IGNORE DONE FLAG.
2022 006426 005737 001126          TST      $BDDAT ;IS CSR CLEAR?
2023 006432 001401          BEQ      7$
2024 006434 104010          ERROR 10          ;AD11K CSR FAILED TO CLEAR.
2025
2026 006436                7$:
2027
2028                ;*      MOV      @ADBUF2,MYTEMP ;/READ DEVICE REG ADBUF2,PUT DATA IN MYTEMP.

```

2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082

*TEST 7 I/O DEVICE TEST-KW11K OPTION

*
*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM) THE
*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
*IT FROM "SRI" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SRI"
*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.

*
* IN THIS TEST WE WILL EXERCISE THE KW11-K. IF
* SRI IS UNALTERED, WE DO THIS TEST SINCE THE
* KW11K IS A DEFAULT I/O DEVICE.
* PLEASE NOTE: THIS IS ONLY A BRIEF TEST OF THE KW11K,
* TO SEE THAT IT ROUGHLY WORKS. FOR TESTING, YOU
* MUST RUN THE KW11-K DIAGNOSTIC. COMPLETE

TESTS:

1. MAKE SURE KW11K RESPONDS TO ADDRESS
(NOTE: IF THE KW11K ON YOUR I/O BUS USES A NON
STANDARD ADDRESS, MODIFY "KWADD" WITH THE CORRECT ADDR.)
2. CLOCK A CSR BITS 14,9,6,3, AND 1 SET
3. CLOCK A CSR BITS 13,8 AND 2 SET
4. CLOCK A CSR ZEROS.
5. CLOCK B CSR BITS 11,6,4 AND 2 SET
6. CLOCK B CSR BITS 5,3, AND 0 SET
7. CLOCK B CSR ZEROS
8. START CLOCK A MODE 0, RATE 1MHZ,
CHECK "ENABLE CNTR A" CLEARS,
"MODE FLAG" AND "A OVERFLOW FLAG" SETS
9. CLOCK A CSR ZEROS
10. START CLOCK B RATE 1MHZ
CHECK "ENB CNTR B" CLEARS, "B OVERFL
FLAG" SETS

*TST7: SCOPE

006446 000004

```

2083 006450 032737 000002 001562 BIT #BIT1,SRI ;IS THIS DEVICE SELECTED FOR TEST?
2084 ;NOTE: DEFAULT=YES.
2085 006456 001002 BNE 10$
2086 006460 000137 007254 JMP 11$
2087 006464 10$:
2088
2089 006464 012737 041110 001124 MOV #BIT14!BIT9!BIT6!BIT3,$GDDAT ;TEST THESE BITS
2090
2091 ;* MOV $GDDAT,2KWADRO ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADRO
2092
2093
2094 006502 005737 037532 TST $AERR ;DID KW11K RESPOND
2095 006506 001403 BEQ 12$ ;TO ADDR? $AERR=0?
2096 ;IF YES-NEXT TEST.
2097 006510 104010 ERROR 10 ;KW11K FAILED TO RESPOND TO ADDR.
2098 006512 000137 007254 JMP 11$
2099 006516 12$:
2100
2101 ;* MOV 2KWADRO,$BDDAT ;/READ DEVICE REG KWADRO,PUT DATA IN $BDDAT.
2102
2103
2104 006526 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID THESE BITS SET?
2105 006534 001403 BEQ 1$
2106
2107 006536 104011 ERROR 11 ;KW11K R/W CSR ERROR.
2108 006540 000137 007254 JMP 11$
2109
2110 006544 012737 020404 001124 1$: MOV #BIT13!BIT8!BIT2,$GDDAT ;GET NEW PATTERN
2111
2112 ;* MOV $GDDAT,2KWADRO ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADRO
2113
2114 ;* MOV 2KWADRO,$BDDAT ;/READ DEVICE REG KWADRO,PUT DATA IN $BDDAT.
2115
2116 006572 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID THE BITS SET?
2117 006600 001403 BEQ 2$
2118
2119 006602 104011 ERROR 11 ;KW11K R/W CSR ERROR.
2120 006604 000137 007254 JMP 11$
2121
2122 006610 005037 001124 2$: CLR $GDDAT ;WRITE ALL ZEROS.
2123
2124 ;* MOV $GDDAT,2KWADRO ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADRO
2125
2126 ;* MOV 2KWADRO,$BDDAT ;/READ DEVICE REG KWADRO,PUT DATA IN $BDDAT.
2127 006634 005737 001126 TST $BDDAT ;DID CSR CLEAR?
2128 006640 001403 BEQ 3$ ;YES-GOOD.
2129
2130 006642 104011 ERROR 11 ;KW11K FAILED TO ZERO CSR.
2131 006644 000137 007254 JMP 11$
2132
2133 006650 012737 004124 001124 3$: MOV #BIT11!BIT6!BIT4!BIT2,$GDDAT ;TEST CLOCK B'S CSR.
2134
2135 ;* MOV $GDDAT,2KWADR4 ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADR4
2136

```

```

2137 ;* MOV @KWADR4,$BDDAT ;/READ DEVICE REG KWADR4,PUT DATA IN $BDDAT.
2138
2139 006676 023737 001126 001124 CMP $BDDAT,$GDDAT ;DID THE BITS SET OK?
2140 006704 001402 BEQ 4$
2141
2142 006706 104011 ERROR 11 ;KW11K CLK B CSR FAILURE.
2143 006710 000561 BR TST10 ;;
2144
2145 006712 012737 000050 001124 4$: MOV #BITS!BIT3,$GDDAT ;TRY THESE BITS
2146
2147
2148 ;* MOV $GDDAT,@KWADR4 ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADR4
2149
2150 ;* MOV @KWADR4,$BDDAT ;/READ DEVICE REG KWADR4,PUT DATA IN $BDDAT.
2151 006740 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID BITS SET OK?
2152 006746 001402 BEQ 5$
2153
2154 006750 104011 ERROR 11 ;KW11K-CLK B CSR ERROR
2155 006752 000540 BR TST10 ;;
2156
2157 006754 005037 001124 5$: CLR $GDDAT ;ZERO CSR
2158
2159 ;* MOV $GDDAT,@KWADR4 ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADR4
2160
2161 ;* MOV @KWADR4,$BDDAT ;/READ DEVICE REG KWADR4,PUT DATA IN $BDDAT.
2162 007000 005737 001126 TST $BDDAT ;DID CSR CLEAR?
2163 007004 001402 BEQ 6$
2164
2165 007006 104011 ERROR 11 ;KW11K-CLKB CSR ERROR.
2166 007010 000521 BR TST10 ;;
2167
2168 007012 012737 000003 001124 6$: MOV #BIT0!BIT1,$GDDAT ;PUT "ENABL CNTR A" AND RATE 1MHZ IN CSR
2169 007020 012737 177777 001126 MOV #177777,$BDDAT ;PRELOAD CNTR.
2170
2171 ;* MOV $BDDAT,@KWADR1 ;/ PUT DATA FROM $BDDAT TO DEVICE REG KWADR1
2172
2173 ;* MOV $GDDAT,@KWADR0 ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADR0
2174
2175 ;* MOV @KWADR0,$BDDAT ;/READ DEVICE REG KWADR0,PUT DATA IN $BDDAT.
2176 ;CLOCK SHOULD OVERFLOW, CLEAR
2177 007056 012737 000242 001124 MOV #BITS!BIT7!BIT1,$GDDAT ;BIT 0 ("ENABLE" CNTR A) SET
2178 ;BIT 5 AND 7
2179 007064 023737 001124 001126 CMP $GDDAT,$BDDAT ;HAPPEN OK?
2180 007072 001401 BEQ 7$
2181 ;KW11K
2182 007074 104011 ERROR 11 ;CLOCK CSR A PROBLEM
2183
2184 007076 005037 001124 7$: CLR $GDDAT ;TRY CLEARING CSR
2185
2186 ;* MOV $GDDAT,@KWADR0 ;/ PUT DATA FROM $GDDAT TO DEVICE REG KWADR0
2187
2188 ;* MOV @KWADR0,$BDDAT ;/READ DEVICE REG KWADR0,PUT DATA IN $BDDAT.
2189 007122 005737 001126 TST $BDDAT ;DID CSR CLEAR?
2190 007126 001402 BEQ 8$

```

```

2191
2192 007130 104011          ERROR 11          ;CSR FAILED TO CLEAR
2193 007132 000450          BR      TST10      ;;
2194
2195 007134 012737 000003 001124 8$:  MOV      #BIT1:BIT0,$GDDAT ;SET "ENABL CNTR B" AND IMHZ IN CSR B
2196 007142 012737 000377 001126  MOV      #377,$BDDAT  ;PRESENT ENTR.
2197
2198          ;*      MOV      $BDDAT,$KWADR5 ; / PUT DATA FROM $BDDAT TO DEVICE REG KWADR5
2199          ;*      MOV      $GDDAT,$KWADR4 ; / PUT DATA FROM $GDDAT TO DEVICE REG KWADR4
2200
2201
2202          ;CLOCK SHOULD OVERFLOW,NOT CLEAR
2203          ;"ENB CNTR B" AND SET
2204          ;"B OVERFL FLAG"
2205
2206          ;*      MOV      $KWADR4,$BDDAT ;/READ DEVICE REG KWADR4,PUT DATA IN $BDDAT.
2207 007200 012737 000203 001124  MOV      #BIT7:BIT1:BIT0,$GDDAT ;EXPECT ONLY OVERFL FLAG SET.
2208 007206 023737 001124 001126  CMP      $GDDAT,$BDDAT ;CSR OK?
2209 007214 001401          BEQ      9$
2210
2211 007216 104011          ERROR 11          ;KW11K CLKB COUNT UP ERROR.
2212
2213
2214 007220 005037 001124          9$:  CLR      $GDDAT      ;TRY CLEARING CSR
2215
2216          ;*      MOV      $GDDAT,$KWADR4 ; / PUT DATA FROM $GDDAT TO DEVICE REG KWADR4
2217          ;*      MOV      $KWADR4,$BDDAT ;/READ DEVICE REG KWADR4,PUT DATA IN $BDDAT.
2218          TST      $BDDAT ;DID CSR CLEAR?
2219 007244 005737 001126          BEQ      TST10      ;;
2220 007250 001401
2221
2222 007252 104011          ERROR 11          ;KW11K CLOCK B SCR FAILED TO CLEAR.
2223
2224 007254          11$:
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244

```

```

*****
*TEST 10          *TEST THE DR11K OPTION,#1, IF "SR1" BIT 2=1(SET)
*
*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE MB254 (IPBM), THE
*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
*
*IN THIS TEST WE WILL CHECK OUT DR11 #1 IF BIT 2 OF SR1=1
*(SET). ALSO, DATA LOOP BACK (OUTPUT DATA BACK TO INPUT) WILL BE CHECKED

```

```

2245 ;*IF "SR2" BIT 2 =1 (SET). FOR THIS, YOU MUST HAVE INSTALLED A LOOP-BACK
2246 ;*CABLE.
2247 ;*****
2248 ST10: SCOPE
2249 007254 000004 000004 001562 BIT #BIT2,SR1
2250 007256 032737 BNE 10$ ;/-TDRT-
2251 007264 001002 JMP 11$
2252 007272 10$:
2253 007272 005037 001514 CLR .DVLS
2254 007276 005037 001124 CLR $GDDAT ;/SET TO xFERR ZEROS.
2255
2256 ;* MOV $GDDAT,DRCR1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR1
2257 007312 005737 037532 TST $AERR ;=1 IF DEVICE NOT PRESENT,
2258 007316 001403 BEQ 1$ ;IF 0, PRESENT, (CON'T).
2259
2260 007320 104012 ERROR 12 ;DR11K #1 DID NOT RESPOND WHEN
2261 ;ADDRESSED. "SR1" BIT 2 SELECTED
2262 ;THIS OPTION. ADDRESS "DRCR"1
2263 ;CONTAINS ITS ADDRESS
2264
2265 007322 000137 007762 JMP 11$
2266
2267 007326 1$:
2268
2269 ;* MOV DRCR1,$BDDAT ;/READ DEVICE REG DRCR1,PUT DATA IN $BDDAT.
2270 007336 005737 001126 TST $BDDAT ;DID CSR CLEAR?
2271 007342 001403 BEQ 2$
2272
2273 007344 104012 ERROR 12 ;DR11K #1 CSR FAILED TO CLEAR.
2274 007346 000137 007762 JMP 11$
2275
2276 007352 2$:
2277
2278 ;* MOV DR0A1,$GDDAT ;/READ DEVICE REG DR0A1,PUT DATA IN $GDDAT.
2279
2280 ;* MOV DR1A1,$GDDAT ;/READ DEVICE REG DR1A1,PUT DATA IN $GDDAT.
2281 007372 012737 040100 001124 MOV #BIT6:BIT14,$GDDAT ;LOAD WRITEABLE BITS.
2282
2283 ;* MOV $GDDAT,DRCR1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR1
2284
2285 ;* MOV DRCR1,$BDDAT ;/READ DEVICE REG DRCR1,PUT DATA IN $BDDAT.
2286 007420 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID BITS SET?
2287 007426 001403 BEQ 3$
2288
2289 007430 104012 ERROR 12 ;DR11K #1 CSR BIT(S) FAILED.
2290 007432 000137 007762 JMP 11$
2291
2292 007436 005037 001124 3$: CLR $GDDAT ;TRY CLEARING CSR.
2293
2294 ;* MOV $GDDAT,DRCR1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR1
2295
2296 ;* MOV DRCR1,$BDDAT ;/READ DEVICE REG DRCR1,PUT DATA IN $BDDAT.
2297 007462 005737 001126 TST $BDDAT ;DID CSR CLEAR?
2298 007466 001403 BEQ 4$ ;YES-NEXT TEST.

```



```

2299
2300 007470 104012          ERROR 12          ;DR11K #1 CSR FAILED TO CLEAR.
2301 007472 000137 007762    JMP     11$
2302
2303 007476 032737 000004 001564 4$:    BIT     #BIT2,SR2    ;DOES THIS DR11 HAVE A LOOP BACK
2304                                ;CABLE CONNECTED TO IT?
2305 007504 001526          BEQ     TST11        ;
2306                                ;IF SR2 BIT2=1 THEN LOOP BACK.
2307
2308 007506 012737 052525 001124    MOV     #052525,$GDDAT ;SET FIRST PATTERN
2309
2310                                ;*    MOV     $GDDAT,DR0A1 ; / PUT DATA FROM $GDDAT TO DEVICE REG DR0A1
2311                                ;*
2312                                ;*    MOV     DR0CR1,$BDDAT ; /READ DEVICE REG DR0CR1,PUT DATA IN $BDDAT.
2313 007534 012737 000200 001124    MOV     #BIT7,$GDDAT ; EXPECT FLAGS TO SET.
2314 007542 023737 001126 001124    CMP     $BDDAT,$GDDAT ; DID THEY?
2315 007550 001402          BEQ     5$          ; YES-CON'T.
2316
2317 007552 104012          ERROR 12          ;DR11K #1 FLAG(S) FAILED TO SET
2318                                ;WHEN DATA XFERRERD. IS THE OUTPUT
2319                                ;REALLY CALLED BACK TO THE INPUT?
2320
2321                                ;
2322 007554 000502          BR      TST11
2323
2324 007556 012737 052525 001124 5$:    MOV     #052525,$GDDAT ; PATTERN THAT SHOULD HAVE XFERRERD.
2325                                ;*
2326 007574 023737 001126 001124    MOV     DR1A1,$BDDAT ; /READ DEVICE REG DR1A1,PUT DATA IN $BDDAT.
2327 007602 001402          CMP     $BDDAT,$GDDAT ; DATA SENT=DATA RECEIVED?
2328                                ;
2329                                ;
2330                                ;
2331                                ;
2332                                ;
2333                                ;
2334 007620 005737 001126    MOV     DR0CR1,$BDDAT ; /READ DEVICE REG DR0CR1,PUT DATA IN $BDDAT.
2335 007624 100402          TST     $BDDAT      ; DID "OUTPUT FLAG" SET.
2336                                ; SHOULD HAVE.
2337
2338 007626 104012          ERROR 12          ;DR11K #1, "OUTPUT FLAG" FAILED TO SET.
2339 007630 000454          BR      TST11
2340
2341                                ;
2342                                ;
2343                                ;
2344                                ;
2345                                ;
2346                                ;
2347                                ;
2348 007670 023737 001126 001124    MOV     $GDDAT,DR1A1 ; / PUT DATA FROM $GDDAT TO DEVICE REG DR1A1
2349 007676 001402          MOV     #125252,$GDDAT ; NEW PATTERN
2350
2351                                ;*
2352                                ;*    MOV     $GDDAT,DR0A1 ; / PUT DATA FROM $GDDAT TO DEVICE REG DR0A1
2353                                ;*
2354                                ;*
2355 007670 023737 001126 001124    MOV     DR1A1,$BDDAT ; /READ DEVICE REG DR1A1,PUT DATA IN $BDDAT.
2356 007676 001402          CMP     $BDDAT,$GDDAT ; PATTERN XFERRER OK?
2357                                ;
2358                                ;
2359                                ;
2360                                ;
2361                                ;
2362                                ;
2363                                ;
2364                                ;
2365                                ;
2366                                ;
2367                                ;
2368                                ;
2369                                ;
2370                                ;
2371                                ;
2372                                ;
2373                                ;
2374                                ;
2375                                ;
2376                                ;
2377                                ;
2378                                ;
2379                                ;
2380                                ;
2381                                ;
2382                                ;
2383                                ;
2384                                ;
2385                                ;
2386                                ;
2387                                ;
2388                                ;
2389                                ;
2390                                ;
2391                                ;
2392                                ;
2393                                ;
2394                                ;
2395                                ;
2396                                ;
2397                                ;
2398                                ;
2399                                ;
2400                                ;
2401                                ;
2402                                ;
2403                                ;
2404                                ;
2405                                ;
2406                                ;
2407                                ;
2408                                ;
2409                                ;
2410                                ;
2411                                ;
2412                                ;
2413                                ;
2414                                ;
2415                                ;
2416                                ;
2417                                ;
2418                                ;
2419                                ;
2420                                ;
2421                                ;
2422                                ;
2423                                ;
2424                                ;
2425                                ;
2426                                ;
2427                                ;
2428                                ;
2429                                ;
2430                                ;
2431                                ;
2432                                ;
2433                                ;
2434                                ;
2435                                ;
2436                                ;
2437                                ;
2438                                ;
2439                                ;
2440                                ;
2441                                ;
2442                                ;
2443                                ;
2444                                ;
2445                                ;
2446                                ;
2447                                ;
2448                                ;
2449                                ;
2450                                ;
2451                                ;
2452                                ;
2453                                ;
2454                                ;
2455                                ;
2456                                ;
2457                                ;
2458                                ;
2459                                ;
2460                                ;
2461                                ;
2462                                ;
2463                                ;
2464                                ;
2465                                ;
2466                                ;
2467                                ;
2468                                ;
2469                                ;
2470                                ;
2471                                ;
2472                                ;
2473                                ;
2474                                ;
2475                                ;
2476                                ;
2477                                ;
2478                                ;
2479                                ;
2480                                ;
2481                                ;
2482                                ;
2483                                ;
2484                                ;
2485                                ;
2486                                ;
2487                                ;
2488                                ;
2489                                ;
2490                                ;
2491                                ;
2492                                ;
2493                                ;
2494                                ;
2495                                ;
2496                                ;
2497                                ;
2498                                ;
2499                                ;
2500                                ;
2501                                ;
2502                                ;
2503                                ;
2504                                ;
2505                                ;
2506                                ;
2507                                ;
2508                                ;
2509                                ;
2510                                ;
2511                                ;
2512                                ;
2513                                ;
2514                                ;
2515                                ;
2516                                ;
2517                                ;
2518                                ;
2519                                ;
2520                                ;
2521                                ;
2522                                ;
2523                                ;
2524                                ;
2525                                ;
2526                                ;
2527                                ;
2528                                ;
2529                                ;
2530                                ;
2531                                ;
2532                                ;
2533                                ;
2534                                ;
2535                                ;
2536                                ;
2537                                ;
2538                                ;
2539                                ;
2540                                ;
2541                                ;
2542                                ;
2543                                ;
2544                                ;
2545                                ;
2546                                ;
2547                                ;
2548                                ;
2549                                ;
2550                                ;
2551                                ;
2552                                ;
2553                                ;
2554                                ;
2555                                ;
2556                                ;
2557                                ;
2558                                ;
2559                                ;
2560                                ;
2561                                ;
2562                                ;
2563                                ;
2564                                ;
2565                                ;
2566                                ;
2567                                ;
2568                                ;
2569                                ;
2570                                ;
2571                                ;
2572                                ;
2573                                ;
2574                                ;
2575                                ;
2576                                ;
2577                                ;
2578                                ;
2579                                ;
2580                                ;
2581                                ;
2582                                ;
2583                                ;
2584                                ;
2585                                ;
2586                                ;
2587                                ;
2588                                ;
2589                                ;
2590                                ;
2591                                ;
2592                                ;
2593                                ;
2594                                ;
2595                                ;
2596                                ;
2597                                ;
2598                                ;
2599                                ;
2600                                ;
2601                                ;
2602                                ;
2603                                ;
2604                                ;
2605                                ;
2606                                ;
2607                                ;
2608                                ;
2609                                ;
2610                                ;
2611                                ;
2612                                ;
2613                                ;
2614                                ;
2615                                ;
2616                                ;
2617                                ;
2618                                ;
2619                                ;
2620                                ;
2621                                ;
2622                                ;
2623                                ;
2624                                ;
2625                                ;
2626                                ;
2627                                ;
2628                                ;
2629                                ;
2630                                ;
2631                                ;
2632                                ;
2633                                ;
2634                                ;
2635                                ;
2636                                ;
2637                                ;
2638                                ;
2639                                ;
2640                                ;
2641                                ;
2642                                ;
2643                                ;
2644                                ;
2645                                ;
2646                                ;
2647                                ;
2648                                ;
2649                                ;
2650                                ;
2651                                ;
2652                                ;
2653                                ;
2654                                ;
2655                                ;
2656                                ;
2657                                ;
2658                                ;
2659                                ;
2660                                ;
2661                                ;
2662                                ;
2663                                ;
2664                                ;
2665                                ;
2666                                ;
2667                                ;
2668                                ;
2669                                ;
2670                                ;
2671                                ;
2672                                ;
2673                                ;
2674                                ;
2675                                ;
2676                                ;
2677                                ;
2678                                ;
2679                                ;
2680                                ;
2681                                ;
2682                                ;
2683                                ;
2684                                ;
2685                                ;
2686                                ;
2687                                ;
2688                                ;
2689                                ;
2690                                ;
2691                                ;
2692                                ;
2693                                ;
2694                                ;
2695                                ;
2696                                ;
2697                                ;
2698                                ;
2699                                ;
2700                                ;
2701                                ;
2702                                ;
2703                                ;
2704                                ;
2705                                ;
2706                                ;
2707                                ;
2708                                ;
2709                                ;
2710                                ;
2711                                ;
2712                                ;
2713                                ;
2714                                ;
2715                                ;
2716                                ;
2717                                ;
2718                                ;
2719                                ;
2720                                ;
2721                                ;
2722                                ;
2723                                ;
2724                                ;
2725                                ;
2726                                ;
2727                                ;
2728                                ;
2729                                ;
2730                                ;
2731                                ;
2732                                ;
2733                                ;
2734                                ;
2735                                ;
2736                                ;
2737                                ;
2738                                ;
2739                                ;
2740                                ;
2741                                ;
2742                                ;
2743                                ;
2744                                ;
2745                                ;
2746                                ;
2747                                ;
2748                                ;
2749                                ;
2750                                ;
2751                                ;
2752                                ;
2753                                ;
2754                                ;
2755                                ;
2756                                ;
2757                                ;
2758                                ;
2759                                ;
2760                                ;
2761                                ;
2762                                ;
2763                                ;
2764                                ;
2765                                ;
2766                                ;
2767                                ;
2768                                ;
2769                                ;
2770                                ;
2771                                ;
2772                                ;
2773                                ;
2774                                ;
2775                                ;
2776                                ;
2777                                ;
2778                                ;
2779                                ;
2780                                ;
2781                                ;
2782                                ;
2783                                ;
2784                                ;
2785                                ;
2786                                ;
2787                                ;
2788                                ;
2789                                ;
2790                                ;
2791                                ;
2792                                ;
2793                                ;
2794                                ;
2795                                ;
2796                                ;
2797                                ;
2798                                ;
2799                                ;
2800                                ;
2801                                ;
2802                                ;
2803                                ;
2804                                ;
2805                                ;
2806                                ;
2807                                ;
2808                                ;
2809                                ;
2810                                ;
2811                                ;
2812                                ;
2813                                ;
2814                                ;
2815                                ;
2816                                ;
2817                                ;
2818                                ;
2819                                ;
2820                                ;
2821                                ;
2822                                ;
2823                                ;
2824                                ;
2825                                ;
2826                                ;
2827                                ;
2828                                ;
2829                                ;
2830                                ;
2831                                ;
2832                                ;
2833                                ;
2834                                ;
2835                                ;
2836                                ;
2837                                ;
2838                                ;
2839                                ;
2840                                ;
2841                                ;
2842                                ;
2843                                ;
2844                                ;
2845                                ;
2846                                ;
2847                                ;
2848                                ;
2849                                ;
2850                                ;
2851                                ;
2852                                ;
2853                                ;
2854                                ;
2855                                ;
2856                                ;
2857                                ;
2858                                ;
2859                                ;
2860                                ;
2861                                ;
2862                                ;
2863                                ;
2864                                ;
2865                                ;
2866                                ;
2867                                ;
2868                                ;
2869                                ;
2870                                ;
2871                                ;
2872                                ;
2873                                ;
2874                                ;
2875                                ;
2876                                ;
2877                                ;
2878                                ;
2879                                ;
2880                                ;
2881                                ;
2882                                ;
2883                                ;
2884                                ;
2885                                ;
2886                                ;
2887                                ;
2888                                ;
2889                                ;
2890                                ;
2891                                ;
2892                                ;
2893                                ;
2894                                ;
2895                                ;
2896                                ;
2897                                ;
2898                                ;
2899                                ;
2900                                ;
2901                                ;
2902                                ;
2903                                ;
2904                                ;
2905                                ;
2906                                ;
2907                                ;
2908                                ;
2909                                ;
2910                                ;
2911                                ;
2912                                ;
2913                                ;
2914                                ;
2915                                ;
2916                                ;
2917                                ;
2918                                ;
2919                                ;
2920                                ;
2921                                ;
2922                                ;
2923                                ;
2924                                ;
2925                                ;
2926                                ;
2927                                ;
2928                                ;
2929                                ;
2930                                ;
2931                                ;
2932                                ;
2933                                ;
2934                                ;
2935                                ;
2936                                ;
2937                                ;
2938                                ;
2939                                ;
2940                                ;
2941                                ;
2942                                ;
2943                                ;
2944                                ;
2945                                ;
2946                                ;
2947                                ;
2948                                ;
2949                                ;
2950                                ;
2951                                ;
2952                                ;
2953                                ;
2954                                ;
2955                                ;
2956                                ;
2957                                ;
2958                                ;
2959                                ;
2960                                ;
2961                                ;
2962                                ;
2963                                ;
2964                                ;
2965                                ;
2966                                ;
2967                                ;
2968                                ;
2969                                ;
2970                                ;
2971                                ;
2972                                ;
2973                                ;
2974                                ;
2975                                ;
2976                                ;
2977                                ;
2978                                ;
2979                                ;
2980                                ;
2981                                ;
2982                                ;
2983                                ;
2984                                ;
2985                                ;
2986                                ;
2987                                ;
2988                                ;
2989                                ;
2990                                ;
2991                                ;
2992                                ;
2993                                ;
2994                                ;
2995                                ;
2996                                ;
2997                                ;
2998                                ;
2999                                ;
3000                                ;

```

```

2353 007704          BS:
2354
2355          ;*      MOV      $GDDAT, @DR1A1  ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR1A1
2356
2357 007714  005037  001124          CLR      $GDDAT          ;NOW XFERR ZERO PATTERN.
2358
2359          ;*      MOV      $GDDAT, @DR0A1  ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR0A1
2360
2361          ;*      MOV      @DR1A1, $BDDAT  ;/READ DEVICE REG DR1A1, PUT DATA IN $BDDAT.
2362 007740  005737  001126          TST      $BDDAT          ;DID ZERO PATTERN GET XFERRED?
2363 007744  001402          BEQ      9$
2364
2365 007746  104012          ERROR    12              ;DR11K #1 DATA XFERR ERROR
2366 007750  000404          BR       TST11          ;;
2367
2368          9$:
2369
2370          ;*      MOV      $GDDAT, @DRCR1  ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR1
2371
2372 007762          11$:
2373
2374          ;*****
2375          ;*TEST 11      *TEST THE DR11K OPTION, #2, IF "SR1" BIT 5=1(SET)
2376          ;*
2377          ;*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
2378          ;*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM) THE
2379          ;*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
2380          ;*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
2381          ;*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
2382          ;*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
2383          ;*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
2384          ;*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
2385          ;*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
2386          ;*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
2387          ;*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
2388          ;*
2389          ;*IN THIS TEST WE WILL CHECK OUT DR11 #2 IF BIT 5 OF SR1-1
2390          ;*(SET). ALSO, DATA LOOP BACK (OUTPUT DATA BACK TO INPUT) WILL BE CHECKED
2391          ;*IF "SR2" BIT 5 =1 (SET). FOR THIS, YOU MUST HAVE INSTALLED A LOOP BACK
2392          ;*CABLE.
2393          ;*****
2394 007762  000004          TST11:  SCOPE
2395 007764  032737  000040  001562          BIT      #BITS, SR1
2396 007772  001002          BNE     10$              ;/-TDRT-
2397 007774  000137  010470          JMP     11$
2398 010000          10$:
2399 010000  005037  001514          CLR     .DVLS
2400 010004  005037  001124          CLR     $GDDAT          ;/SET TO XFERR ZEROS.
2401
2402          ;*      MOV      $GDDAT, @DRCR2  ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR2
2403 010020  005737  037532          TST     $AERR           ;=1 IF DEVICE NOT PRESENT,
2404 010024  001403          BEQ     1$              ;IF 0, PRESENT, (CON'T).
2405
2406 010026  104012          ERROR    12              ;DR11K #2 DID NOT RESPOND WHEN

```

```

2407 ;ADDRESSED. "SR1" BIT 5 SELECTED
2408 ;THIS OPTION. ADDRESS "DRCR"2
2409 ;CONTAINS ITS ADDRESS
2410
2411 010030 000137 010470 JMP 11$
2412
2413 010034 1$:
2414
2415 ;* MOV @DRCR2,$BDDAT ;/READ DEVICE REG DRCR2,PUT DATA IN $BDDAT.
2416 TST $BDDAT ;DID CSR CLEAR?
2417 010044 005737 001126 BEQ 2$
2418 010050 001403
2419
2420 010052 104012 ERROR 12 ;DR11K #2 CSR FAILED TO CLEAR.
2421 010054 000137 010470 JMP 11$
2422
2423 010060 2$:
2424
2425 ;* MOV @DROA2,$GDDAT ;/READ DEVICE REG DROA2,PUT DATA IN $GDDAT.
2426
2427 ;* MOV @DRIA2,$GDDAT ;/READ DEVICE REG DRIA2,PUT DATA IN $GDDAT.
2428 010100 012737 040100 001124 MOV #BIT6!BIT14,$GDDAT ;LOAD WRITEABLE BITS.
2429
2430 ;* MOV $GDDAT,@DRCR2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR2
2431
2432 ;* MOV @DRCR2,$BDDAT ;/READ DEVICE REG DRCR2,PUT DATA IN $BDDAT.
2433 010126 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID BITS SET?
2434 010134 001403 BEQ 3$
2435
2436 010136 104012 ERROR 12 ;DR11K #2 CSR BIT(S) FAILED.
2437 010140 000137 010470 JMP 11$
2438
2439 010144 005037 001124 3$: CLR $GDDAT ;TRY CLEARING CSR.
2440
2441 ;* MOV $GDDAT,@DRCR2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR2
2442
2443 ;* MOV @DRCR2,$BDDAT ;/READ DEVICE REG DRCR2,PUT DATA IN $BDDAT.
2444 010170 005737 001126 TST $BDDAT ;DID CSR CLEAR?
2445 010174 001403 BEQ 4$ ;YES-NEXT TEST.
2446
2447 010176 104012 ERROR 12 ;DR11K #2 CSR FAILED TO CLEAR.
2448 010200 000137 010470 JMP 11$
2449
2450 010204 032737 000040 001564 4$: BIT #BITS,SR2 ;DOES THIS DR11 HAVE A LOOP BACK
2451 010212 001526 BEQ TST12 ;CABLE CONNECTED TO IT?
2452 ;IF SR2 BITS=1 THEN LOOP BACK.
2453
2454 010214 012737 052525 001124 MOV #052525,$GDDAT ;SET FIRST PATTERN
2455
2456 ;* MOV $GDDAT,@DROA2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DROA2
2457
2458 ;* MOV @DRCR2,$BDDAT ;/READ DEVICE REG DRCR2,PUT DATA IN $BDDAT.
2459 010242 012737 000200 001124 MOV #BIT7,$GDDAT ;EXPECT FLAGS TO SET.
2460 010250 023737 001126 001124 CMP $BDDAT,$GDDAT ;DID THEY?

```

```

2461 010256 001402          BEQ      5$          ;YES-CON'T.
2462
2463 010260 104012          ERROR    12          ;DR11K #2 FLAG(S) FAILED TO SET
2464                                ;WHEN DATA XFERRD. IS THE OUTPUT
2465                                ;REALLY CALLED BACK TO THE INPUT?
2466 010262 000502          BR       TST12       ;
2467
2468 010264 012737 052525 001124 5$:  MOV      #052525,$GDDAT ;PATTERN THAT SHOULD HAVE XFERRD.
2469
2470                                ;*
2471 010302 023737 001126 001124 ;*  MOV      @DRIA2,$BDDAT ;/READ DEVICE REG DRIA2,PUT DATA IN $BDDAT.
2472 010310 001402          CMP      $BDDAT,$GDDAT ;DATA SENT=DATA RECEIVED?
2473                                BEQ      6$
2474 010312 104012          ERROR    12          ;DR11K #2 DATA XFERR ERROR.
2475 010314 000465          BR       TST12       ;
2476
2477 010316                                6$:
2478                                ;*
2479                                ;*
2480 010326 005737 001126 ;*  MOV      @DRCR2,$BDDAT ;/READ DEVICE REG DRCR2,PUT DATA IN $BDDAT.
2481 010332 100402          TST     $BDDAT       ;DID "OUTPUT FLAG" SET.
2482                                BMI      7$          ;SHOULD HAVE.
2483 010334 104012          ERROR    12          ;DR11K #2, "OUTPUT FLAG" FAILED TO SET.
2484 010336 000454          BR       TST12       ;
2485
2486 010340                                7$:
2487
2488                                ;*
2489 010350 012737 125252 001124 ;*  MOV      $GDDAT,@DRIA2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRIA2
2490                                MOV      #125252,$GDDAT ;NEW PATTERN
2491                                ;*
2492                                ;*
2493                                ;*
2494 010376 023737 001126 001124 ;*  MOV      @DRIA2,$BDDAT ;/READ DEVICE REG DRIA2,PUT DATA IN $BDDAT.
2495 010404 001402          CMP      $BDDAT,$GDDAT ;PATTERN XFERR OK?
2496                                BEQ      8$
2497 010406 104012          ERROR    12          ;DR11K #2 DATA XFERR ERROR
2498 010410 000427          BR       TST12       ;
2499 010412                                8$:
2500
2501                                ;*
2502                                ;*
2503 010422 005037 001124 ;*  MOV      $GDDAT,@DRIA2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRIA2
2504                                CLR      $GDDAT       ;NOW XFERR ZERO PATTERN.
2505                                ;*
2506                                ;*
2507                                ;*
2508 010446 005737 001126 ;*  MOV      @DRIA2,$BDDAT ;/READ DEVICE REG DRIA2,PUT DATA IN $BDDAT.
2509 010452 001402          TST     $BDDAT       ;DID ZERO PATTERN GET XFERRD?
2510                                BEQ      9$
2511 010454 104012          ERROR    12          ;DR11K #2 DATA XFERR ERROR
2512 010456 000404          BR       TST12       ;
2513
2514 010460                                9$:

```

```

2515 ;*      MOV      $GDDAT,DRCR2    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR2
2516
2517
2518 010470 11$:
2519
2520 ;*****
2521 ;*TEST 12      *TEST THE DR11K OPTION,#3, IF "SR1" BIT 7=1(SET)
2522 ;*
2523 ;*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
2524 ;*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
2525 ;*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
2526 ;*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
2527 ;*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
2528 ;*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
2529 ;*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
2530 ;*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
2531 ;*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
2532 ;*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
2533 ;*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
2534 ;*
2535 ;*IN THIS TEST WE WILL CHECK OUT DR11 #3 IF BIT 7 OF SR1=1
2536 ;*(SET). ALSO, DATA LOOP BACK (OUTPUT DATA BACK TO INPUT) WILL BE CHECKED
2537 ;*IF "SR2" BIT 7 =1 (SET). FOR THIS, YOU MUST HAVE INSTALLED A LOOP-BACK
2538 ;*CABLE.
2539 ;*****
2540 010470 000004
2541 010472 032737 000200 001562
2542 010500 001002
2543 010502 000137 011176
2544 010506
2545 010506 005037 001514
2546 010512 005037 001124
2547
2548 ;*      MOV      $GDDAT,DRCR3    ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR3
2549 010526 005737 037532
2550 010532 001403
2551
2552 010534 104012
2553
2554
2555 ;DR11K #3 DID NOT RESPOND WHEN
2556 ;ADDRESSED. "SR1" BIT 7 SELECTED
2557 ;THIS OPTION. ADDRESS "DRCR"3
2558 ;CONTAINS ITS ADDRESS
2559
2560
2561 010536 000137 011176
2562
2563 010542
2564
2565 ;*      MOV      DRRCR3,$BDDAT   ;/READ DEVICE REG DRCR3,PUT DATA IN $BDDAT.
2566 010552 005737 001126
2567 010556 001403
2568
2569 ;DID CSR CLEAR?
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700

```

```

2569
2570 ;* MOV @DROA3,$GDDAT ;/READ DEVICE REG DROA3,PUT DATA IN $GDDAT.
2571 ;* MOV @DRIA3,$GDDAT ;/READ DEVICE REG DRIA3,PUT DATA IN $GDDAT.
2572 010606 012737 040100 001124 ;* MOV #BIT6!BIT14,$GDDAT ;LOAD WRITEABLE BITS.
2573
2574 ;* MOV $GDDAT,@DRCR3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR3
2575
2576 ;* MOV @DRCR3,$BDDAT ;/READ DEVICE REG DRCR3,PUT DATA IN $BDDAT.
2577 010634 023737 001124 001126 ;* CMP $GDDAT,$BDDAT ;DID BITS SET?
2578 010642 001403 BEQ 3$
2579
2580 ERROR 12 ;DR11K #3 CSR BIT(S) FAILED.
2581 010644 104012 JMP 11$
2582 010646 000137 011176
2583
2584 010652 005037 001124 3$: CLR $GDDAT ;TRY CLEARING CSR.
2585
2586 ;* MOV $GDDAT,@DRCR3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR3
2587
2588 ;* MOV @DRCR3,$BDDAT ;/READ DEVICE REG DRCR3,PUT DATA IN $BDDAT.
2589 010676 005737 001126 ;* TST $BDDAT ;DID CSR CLEAR?
2590 010702 001403 BEQ 4$ ;YES-NEXT TEST.
2591
2592 ERROR 12 ;DR11K #3 CSR FAILED TO CLEAR.
2593 010704 104012 JMP 11$
2594 010706 000137 011176
2595
2596 010712 032737 000200 001564 4$: BIT #BIT7,SR2 ;DOES THIS DR11 HAVE A LOOP BACK
2597 010720 001526 BEQ TST13 ;CABLE CONNECTED TO IT?
2598 ;IF SR2 BIT7=1 THEN LOOP BACK.
2599
2600 010722 012737 052525 001124 MOV #052525,$GDDAT ;SET FIRST PATTERN
2601
2602 ;* MOV $GDDAT,@DROA3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DROA3
2603
2604 ;* MOV @DRCR3,$BDDAT ;/READ DEVICE REG DRCR3,PUT DATA IN $BDDAT.
2605 010750 012737 000200 001124 ;* MOV #BIT7,$GDDAT ;EXPECT FLAGS TO SET.
2606 010756 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;DID THEY?
2607 010764 001402 BEQ 5$ ;YES-CON'T.
2608
2609 ERROR 12 ;DR11K #3 FLAG(S) FAILED TO SET
2610 ;WHEN DATA XFERRERD. IS THE OUTPUT
2611 ;REALLY CALLED BACK TO THE INPUT?
2612 010770 000502 BR TST13 ;
2613
2614 010772 012737 052525 001124 5$: MOV #052525,$GDDAT ;PATTERN THAT SHOULD HAVE XFERRERD.
2615
2616 ;* MOV @DRIA3,$BDDAT ;/READ DEVICE REG DRIA3,PUT DATA IN $BDDAT.
2617 011010 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;DATA SENT=DATA RECEIVED?
2618 011016 001402 BEQ 6$
2619
2620 011020 104012 ERROR 12 ;DR11K #3 DATA XFERR ERROR.
2621 011022 000465 BR TST13 ;
2622

```

```

2623 011024 6S:
2624
2625 ;* MOV 2DRCR3,$BDDAT ;/READ DEVICE REG DRCR3,PUT DATA IN $BDDAT.
2626 011034 005737 001126 ;* TST $BDDAT ;DID "OUTPUT FLAG" SET.
2627 011040 100402 ;* BMI 7S ;SHOULD HAVE.
2628
2629 011042 104012 ;* ERROR 12 ;DR11K #3, "OUTPUT FLAG" FAILED TO SET.
2630 011044 000454 ;* BR TST13 ;
2631
2632 011046 7S:
2633
2634 ;* MOV $GDDAT,2DRIA3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRIA3
2635 011056 012737 125252 001124 ;* MOV #125252,$GDDAT ;NEW PATTERN
2636
2637 ;* MOV $GDDAT,2DROA3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DROA3
2638
2639 ;* MOV 2DRIA3,$BDDAT ;/READ DEVICE REG DRIA3,PUT DATA IN $BDDAT.
2640 011104 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;PATTERN XFERR OK?
2641 011112 001402 ;* BEQ 8S
2642
2643 011114 104012 ;* ERROR 12 ;DR11K #3 DATA XFERR ERROR
2644 011116 000427 ;* BR TST13 ;
2645
2646 8S:
2647
2648 ;* MOV $GDDAT,2DRIA3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRIA3
2649 011130 005037 001124 ;* CLR $GDDAT ;NOW XFERR ZERO PATTERN.
2650
2651 ;* MOV $GDDAT,2DROA3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DROA3
2652
2653 ;* MOV 2DRIA3,$BDDAT ;/READ DEVICE REG DRIA3,PUT DATA IN $BDDAT.
2654 011154 005737 001126 ;* TST $BDDAT ;DID ZERO PATTERN GET XFERRERD?
2655 011160 001402 ;* BEQ 9S
2656
2657 011162 104012 ;* ERROR 12 ;DR11K #3 DATA XFERR ERROR
2658 011164 000404 ;* BR TST13 ;
2659
2660 011166 9S:
2661
2662 ;* MOV $GDDAT,2DRCR3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR3
2663
2664 011176 11S:
2665
2666 ;*****
2667 ;*TEST 13 *TEST THE DR11K OPTION,#4, IF "SR1" BIT 8=1(SET)
2668 ;*
2669 ;*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
2670 ;*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
2671 ;*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
2672 ;*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
2673 ;*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
2674 ;*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
2675 ;*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
2676 ;*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE

```

```

2677 ;*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
2678 ;*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
2679 ;*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
2680 ;*
2681 ;*IN THIS TEST WE WILL CHECK OUT DR11 #4 IF BIT 8 OF SR1=1
2682 ;*(SET). ALSO, DATA LOOP BACK (OUTPUT DATA BACK TO INPUT) WILL BE CHECKED
2683 ;*IF "SR2" BIT 8 =1 (SET). FOR THIS, YOU MUST HAVE INSTALLED A LOOP-BACK
2684 ;*CABLE.
2685 ;*****
2686 011176 000004          TST13: SCOPE
2687 011200 032737 000400 001562 BIT      #BIT8,SR1
2688 011206 001002          BNE      10$      ;/-TDRT-
2689 011210 000137 011704 JMP      11$
2690 011214          10$: CLR      .DVLS
2691 011214 005037 001514 CLR      $GDDAT      ;/SET TO XFERR ZEROS.
2692 011220 005037 001124
2693
2694 ;* MOV      $GDDAT,DRCR4      ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR4
2695 011234 005737 037532 TST      $AERR      ;=1 IF DEVICE NOT PRESENT,
2696 011240 001403 BEQ      1$      ;IF 0, PRESENT, (CON'T).
2697
2698 011242 104012          ERROR  12      ;DR11K #4 DID NOT RESPOND WHEN
2699          ;ADDRESSED. "SR1" BIT 8 SELECTED
2700          ;THIS OPTION. ADDRESS "DRCR"4
2701          ;CONTAINS ITS ADDRESS
2702
2703 011244 000137 011704 JMP      11$
2704
2705 011250          1$:
2706
2707 ;* MOV      DRCR4,$BDDAT      ;/READ DEVICE REG DRCR4,PUT DATA IN $BDDAT.
2708 011260 005737 001126 TST      $BDDAT      ;DID CSR CLEAR?
2709 011264 001403 BEQ      2$
2710
2711 011266 104012          ERROR  12      ;DR11K #4 CSR FAILED TO CLEAR.
2712 011270 000137 011704 JMP      11$
2713
2714 011274          2$:
2715
2716 ;* MOV      DROR4,$GDDAT      ;/READ DEVICE REG DRO4,PUT DATA IN $GDDAT.
2717
2718 ;* MOV      DRIR4,$GDDAT      ;/READ DEVICE REG DRIR4,PUT DATA IN $GDDAT.
2719 011314 012737 040100 001124 MOV      #BIT6:BIT14,$GDDAT ;LOAD WRITEABLE BITS.
2720
2721 ;* MOV      $GDDAT,DRCR4      ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR4
2722
2723 ;* MOV      DRCR4,$BDDAT      ;/READ DEVICE REG DRCR4,PUT DATA IN $BDDAT.
2724 011342 023737 001124 001126 CMP      $GDDAT,$BDDAT ;DID BITS SET?
2725 011350 001403 BEQ      3$
2726
2727 011352 104012          ERROR  12      ;DR11K #4 CSR BIT(S) FAILED.
2728 011354 000137 011704 JMP      11$
2729
2730 011360 005037 001124 3$: CLR      $GDDAT      ;TRY CLEARING CSR.

```



```

2731
2732          ;*      MOV      $GDDAT,DRCR4      ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR4
2733
2734          ;*      MOV      DRCR4,$BDDAT      ;/READ DEVICE REG DRCR4,PUT DATA IN $BDDAT.
2735 011404 005737 001126          TST      $BDDAT      ;DID CSR CLEAR?
2736 011410 001403          BEQ      4$          ;YES-NEXT TEST.
2737
2738 011412 104012          ERROR   12          ;DR11K #4 CSR FAILED TO CLEAR.
2739 011414 000137 011704          JMP      11$
2740
2741 011420 032737 000400 001564 4$:  BIT      #BIT8,SR2      ;DOES THIS DR11 HAVE A LOOP BACK
2742          BEQ      TST14      ;CABLE CONNECTED TO IT?
2743 011426 001526          ;
2744          ;IF SR2 BIT8=1 THEN LOOP BACK.
2745
2746 011430 012737 052525 001124      MOV      #052525,$GDDAT ;SET FIRST PATTERN
2747
2748          ;*      MOV      $GDDAT,DROR4      ;/ PUT DATA FROM $GDDAT TO DEVICE REG DROA4
2749
2750          ;*      MOV      DRCR4,$BDDAT      ;/READ DEVICE REG DRCR4,PUT DATA IN $BDDAT.
2751 011456 012737 000200 001124      MOV      #BIT7,$GDDAT      ;EXPECT FLAGS TO SET.
2752 011464 023737 001126 001124      CMP      $BDDAT,$GDDAT      ;DID THEY?
2753 011472 001402          BEQ      5$          ;YES-CON'T.
2754
2755 011474 104012          ERROR   12          ;DR11K #4 FLAG(S) FAILED TO SET
2756          ;WHEN DATA XFERRD. IS THE OUTPUT
2757          ;REALLY CALLED BACK TO THE INPUT?
2758 011476 000502          BR      TST14          ;
2759
2760 011500 012737 052525 001124 5$:  MOV      #052525,$GDDAT ;PATTERN THAT SHOULD HAVE XFERRD.
2761
2762          ;*      MOV      DR1A4,$BDDAT      ;/READ DEVICE REG DR1A4,PUT DATA IN $BDDAT.
2763 011516 023737 001126 001124      CMP      $BDDAT,$GDDAT      ;DATA SENT=DATA RECEIVED?
2764 011524 001402          BEQ      6$
2765
2766 011526 104012          ERROR   12          ;DR11K #4 DATA XFERR ERROR.
2767 011530 000465          BR      TST14          ;
2768
2769 011532          6$:
2770
2771          ;*      MOV      DRCR4,$BDDAT      ;/READ DEVICE REG DRCR4,PUT DATA IN $BDDAT.
2772 011542 005737 001126          TST      $BDDAT      ;DID "OUTPUT FLAG" SET.
2773 011546 100402          BMI      7$          ;SHOULD HAVE.
2774
2775 011550 104012          ERROR   12          ;DR11K #4, "OUTPUT FLAG" FAILED TO SET.
2776 011552 000454          BR      TST14          ;
2777
2778 011554          7$:
2779
2780          ;*      MOV      $GDDAT,DR1A4      ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR1A4
2781 011564 012737 125252 001124      MOV      #125252,$GDDAT ;NEW PATTERN
2782
2783          ;*      MOV      $GDDAT,DROR4      ;/ PUT DATA FROM $GDDAT TO DEVICE REG DROA4
2784

```

```

2785      ;*      MOV      @DRI4,$BDDAT ;/READ DEVICE REG DRI4,PUT DATA IN $BDDAT.
2786 011612 023737 001126 001124 ;*      CMP      $BDDAT,$GDDAT ;PATTERN XFERR OK?
2787 011620 001402      ;*      BEQ      8$
2788
2789 011622 104012      ;*      ERROR   12      ;DR11K #4 DATA XFERR ERROR
2790 011624 000427      ;*      BR       TST14      ;;
2791 011626      ;*      8$:
2792
2793      ;*      MOV      $GDDAT,@DRI4 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRI4
2794
2795 011636 005037 001124      ;*      CLR      $GDDAT ;NOW XFERR ZERO PATTERN.
2796
2797      ;*      MOV      $GDDAT,@DRO4 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRO4
2798
2799      ;*      MOV      @DRI4,$BDDAT ;/READ DEVICE REG DRI4,PUT DATA IN $BDDAT.
2800 011662 005737 001126      ;*      TST      $BDDAT ;DID ZERO PATTERN GET XFERRERD?
2801 011666 001402      ;*      BEQ      9$
2802
2803 011670 104012      ;*      ERROR   12      ;DR11K #4 DATA XFERR ERROR
2804 011672 000404      ;*      BR       TST14      ;;
2805
2806 011674      ;*      9$:
2807
2808      ;*      MOV      $GDDAT,@DRCR4 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR4
2809
2810 011704      ;*      11$:
2811
2812      ;*****
2813      ;*TEST 14      *TEST THE DR11K OPTION,#5, IF "SR1" BIT 9=1(SET)
2814      ;*
2815      ;*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
2816      ;*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
2817      ;*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
2818      ;*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
2819      ;*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
2820      ;*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
2821      ;*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
2822      ;*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
2823      ;*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
2824      ;*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
2825      ;*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
2826      ;*
2827      ;*IN THIS TEST WE WILL CHECK OUT DR11 #5 IF BIT 9 OF SR1=1
2828      ;*(SET). ALSO, DATA LOOP BACK (OUTPUT DATA BACK TO INPUT) WILL BE CHECKED
2829      ;*IF "SR2" BIT 9 =1 (SET). FOR THIS, YOU MUST HAVE INSTALLED A LOOP-BACK
2830      ;*CABLE.
2831      ;*****
2832 011704 000004      ;*ST14: SCOPE
2833 011706 032737 001000 001562      ;*      BIT      #BIT9,SR1
2834 011714 001002      ;*      BNE     10$      ,/-TDRT-
2835 011716 000137 012412      ;*      JMP     11$
2836 011722      ;*      10$:
2837 011722 005037 001514      ;*      CLR     .DVLS
2838 011726 005037 001124      ;*      CLR     $GDDAT ;/SET TO XFERR ZEROS.

```

```

2840 011742 005737 037532 ; MOV $GDDAT,DRCR5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCRS
2841 011746 001403 TST $AERR ;=1 IF DEVICE NOT PRESENT,
2842 011750 104012 BEQ 1$ ;IF 0, PRESENT, (CON'T).
2843 011752 000137 012412 ERROR 12 ;DR11K #5 DID NOT RESPOND WHEN
2844 011756 000137 012412 ;ADDRESSED. "SR1" BIT 9 SELECTED
2845 011774 104012 ;THIS OPTION. ADDRESS "DRCR"5
2846 011776 000137 012412 JMP 11$ ;CONTAINS ITS ADDRESS
2847 011774 104012 ERROR 12 ;DR11K #5 CSR FAILED TO CLEAR.
2848 011776 000137 012412 JMP 11$
2849 012002 2$:
2850 012022 012737 040100 001124 ;* MOV DRDRA5,$GDDAT ;/READ DEVICE REG DRDRA5,PUT DATA IN $GDDAT.
2851 012022 012737 040100 001124 ;* MOV DRDRIAS,$GDDAT ;/READ DEVICE REG DRDRIAS,PUT DATA IN $GDDAT.
2852 012022 012737 040100 001124 ;* MOV #BIT6:BIT14,$GDDAT ;LOAD WRITEABLE BITS.
2853 012022 012737 040100 001124 ;* MOV $GDDAT,DRCR5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCRS
2854 012050 023737 001124 001126 ;* MOV DRDRCR5,$BDDAT ;/READ DEVICE REG DRCRS,PUT DATA IN $BDDAT.
2855 012056 001403 001124 001126 ;* CMP $GDDAT,$BDDAT ;DID BITS SET?
2856 012056 001403 001124 001126 ;* BEQ 3$
2857 012060 104012 ERROR 12 ;DR11K #5 CSR BIT(S) FAILED.
2858 012062 000137 012412 JMP 11$
2859 012066 005037 001124 3$: CLR $GDDAT ;TRY CLEARING CSR.
2860 012066 005037 001124 3$: ;
2861 012066 005037 001124 3$: ;* MOV $GDDAT,DRCR5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCRS
2862 012112 005737 001126 ;* MOV DRDRCR5,$BDDAT ;/READ DEVICE REG DRCRS,PUT DATA IN $BDDAT.
2863 012116 001403 001126 ;* TST $BDDAT ;DID CSR CLEAR?
2864 012120 104012 012412 ;* BEQ 4$ ;YES-NEXT TEST.
2865 012120 104012 012412 ;* ERROR 12 ;DR11K #5 CSR FAILED TO CLEAR.
2866 012122 000137 012412 ;* JMP 11$
2867 012126 032737 001000 001564 4$: BIT #BIT9,SR2 ;DOES THIS DR11 HAVE A LOOP BACK
2868 012134 001526 001564 4$: BEQ TST15 ;CABLE CONNECTED TO IT?
2869 012136 012737 052525 001124 4$: ;IF SR2 BIT9=1 THEN LOOP BACK.
2870 012136 012737 052525 001124 4$: MOV #052525,$GDDAT ;SET FIRST PATTERN

```

```

2893
2894 ;* MOV $GDDAT,DR0A5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR0A5
2895 ;* MOV DR0CR5,$BDDAT ;/READ DEVICE REG DR0CR5,PUT DATA IN $BDDAT.
2896 ;* MOV DR0CR5,$BDDAT ;/READ DEVICE REG DR0CR5,PUT DATA IN $BDDAT.
2897 012164 012737 000200 001124 MOV #BIT7,$GDDAT ;EXPECT FLAGS TO SET.
2898 012172 023737 001126 001124 CMP $BDDAT,$GDDAT ;DID THEY?
2899 012200 001402 BEQ 5$ ;YES-CON'T.
2900
2901 012202 104012 ERROR 12 ;DR11K #5 FLAG(S) FAILED TO SET
2902 ;WHEN DATA XFERRERD. IS THE OUTPUT
2903 ;REALLY CALLED BACK TO THE INPUT?
2904 012204 000502 BR TST15 ;
2905
2906 012206 012737 052525 001124 5$: MOV #052525,$GDDAT ;PATTERN THAT SHOULD HAVE XFERRERD.
2907
2908 ;* MOV DR1A5,$BDDAT ;/READ DEVICE REG DR1A5,PUT DATA IN $BDDAT.
2909 012224 023737 001126 001124 CMP $BDDAT,$GDDAT ;DATA SENT=DATA RECEIVED?
2910 012232 001402 BEQ 6$
2911
2912 012234 104012 ERROR 12 ;DR11K #5 DATA XFERR ERROR.
2913 012236 000465 BR TST15 ;
2914
2915 012240 6$:
2916
2917 ;* MOV DR0CR5,$BDDAT ;/READ DEVICE REG DR0CR5,PUT DATA IN $BDDAT.
2918 012250 005737 001126 TST $BDDAT ;DID "OUTPUT FLAG" SET.
2919 012254 100402 BMI 7$ ;SHOULD HAVE.
2920
2921 012256 104012 ERROR 12 ;DR11K #5, "OUTPUT FLAG" FAILED TO SET.
2922 012260 000454 BR TST15 ;
2923
2924 012262 7$:
2925
2926 ;* MOV $GDDAT,DR1A5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR1A5
2927 012272 012737 125252 001124 MOV #125252,$GDDAT ;NEW PATTERN
2928
2929 ;* MOV $GDDAT,DR0A5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR0A5
2930
2931 ;* MOV DR1A5,$BDDAT ;/READ DEVICE REG DR1A5,PUT DATA IN $BDDAT.
2932 012320 023737 001126 001124 CMP $BDDAT,$GDDAT ;PATTERN XFERR OK?
2933 012326 001402 BEQ 8$
2934
2935 012330 104012 ERROR 12 ;DR11K #5 DATA XFERR ERROR
2936 012332 000427 BR TST15 ;
2937 012334 8$:
2938
2939 ;* MOV $GDDAT,DR1A5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR1A5
2940
2941 012344 005037 001124 CLR $GDDAT ;NOW XFERR ZERO PATTERN.
2942
2943 ;* MOV $GDDAT,DR0A5 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DR0A5
2944
2945 ;* MOV DR1A5,$BDDAT ;/READ DEVICE REG DR1A5,PUT DATA IN $BDDAT.
2946 012370 005737 001126 TST $BDDAT ;DID ZERO PATTERN GET XFERRERD?

```

JOB

MAINDEC -11- DRLPA-A
DRLPA.P11 T14

MACY11 27(654) 13-DEC-77 12:58 PAGE 70
*TEST THE DRI1K OPTION, #5, IF "SR1" BIT 9=1(SET)

SEQ 0100

```

2947 012374 001402          BEQ      9$
2948
2949 012376 104012          ERROR   12          ;DRI1K #5 DATA XFERR ERROR
2950 012400 000404          BR       TST15          ;;
2951
2952 012402          9$:
2953
2954          ;*      MOV      $GDDAT,DRCR5  ;/ PUT DATA FROM $GDDAT TO DEVICE REG DRCR5
2955
2956 012412          11$:
2957
2958
2959          ;*****
2960          ;*TEST 15          *TEST THE AA-11 OPTION, IF "SR1" BIT 3=1 (SET)
2961          ;*
2962          ;*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
2963          ;*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE MB254 (IPBM), THE
2964          ;*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
2965          ;*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
2966          ;*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
2967          ;*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
2968          ;*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
2969          ;*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
2970          ;*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
2971          ;*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
2972          ;*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
2973          ;*
2974          ;*IN THIS TEST WE WILL CHECK OUT THE AA-11 OPTION,
2975          ;*IF BIT 3 OF "SR1" IS SET.
2976          ;*
2977          ;*****
2978 012412 000004          †TST15: SCOPE
2979
2980 012414 032737 000010 001562          BIT      #BIT3,SR1          ;IS AA-11 SELECTED
2981 012422 001002          BNE      20$          ;IF =0, NO, SKIP THIS TEST.
2982 012424 000137 013442          JMP      19$
2983 012430          20$:
2984 012430 005037 001124          CLR      $GDDAT
2985
2986          ;*      MOV      $GDDAT,DRACSR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARCSR
2987 012444 005737 037532          TST     $AERR          ;DID WE SUCCESSFULLY WRITE THE AA-11?
2988 012450 001403          BEQ     1$          ;YES IF $AERR=0
2989
2990 012452 104014          ERROR   14          ;FAILED TO ADDRESS AA-11. "SR1" BIT 3=1
2991          ;SELECTED THIS TEST. AA-11 ADDRESS IS
2992          ;IN LOC "ARCSR"
2993 012454 000137 013442          JMP     19$
2994
2995 012460          1$:
2996
2997          ;*      MOV     DRACSR,$BDDAT  ;/READ DEVICE REG ARCSR,PUT DATA IN $BDDAT.
2998 012470 042737 000200 001126          BIC     #BIT7,$BDDAT  ;CLEAR FLAG
2999 012476 001403          BEQ     2$          ;DID CSR CLEAR?
3000

```

```

3001 012500 104014          ERROR 14          ;CLEAR CSR FAILED (AA-11K)
3002 012502 000137 013442    JMP    19$
3003
3004 012506 012737 005124 001124 2$:  MOV    #5124,$GDDAT ;TRY WRITING THESE BITS
3005
3006          ;*      MOV    $GDDAT,$AACSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG AACSR
3007
3008          ;*      MOV    $AACSR,$BDDAT ;/READ DEVICE REG AACSR,PUT DATA IN $BDDAT.
3009          ;      ;READ THEM BACK
3010 012534 042737 000200 001126    BIC    #BIT7,$BDDAT ;GET RID OF FLAG
3011 012542 023737 001124 001126    CMP    $GDDAT,$BDDAT ;DID PATTERN W/R OK?
3012 012550 001403          BEQ    3$          ;YES-NEXT TEST.
3013
3014 012552 104014          ERROR 14          ;AA11-K PATTERN 5124 FAILED TO W/R PROPERLY.
3015 012554 000137 013442    JMP    19$
3016
3017 012560 012737 002012 001124 3$:  MOV    #2012,$GDDAT ;TRY NEW PATTERN.
3018
3019          ;*      MOV    $GDDAT,$AACSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG AACSR
3020
3021          ;*      MOV    $AACSR,$BDDAT ;/READ DEVICE REG AACSR,PUT DATA IN $BDDAT.
3022 012606 042737 000200 001126    BIC    #BIT7,$BDDAT ;GET RID OF FLAG.
3023 012614 023737 001124 001126    CMP    $GDDAT,$BDDAT ;DID PATTERN XFERR OK?
3024 012622 001403          BEQ    4$          ;YES-NEXT TEST.
3025
3026 012624 104014          ERROR 14          ;AA-11K PATTERN 2012 FAILED TO W/R PROPERLY.
3027 012626 000137 013442    JMP    19$
3028
3029 012632 005037 001124          4$:  CLR    $GDDAT      ;TRY TO CLEAR CSR.
3030
3031          ;*      MOV    $GDDAT,$AACSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG AACSR
3032
3033          ;*      MOV    $AACSR,$BDDAT ;/READ DEVICE REG AACSR,PUT DATA IN $BDDAT.
3034 012656 042737 000200 001126    BIC    #BIT7,$BDDAT ;CLEAR FLAG BIT.
3035 012664 001403          BEQ    5$
3036
3037 012666 104014          ERROR 14          ;AA-11K CSR FAILED TO CLEAR.
3038 012670 000137 013442    JMP    19$
3039
3040 012674 012737 005252 001124 5$:  MOV    #5252,$GDDAT
3041
3042          ;*      MOV    $GDDAT,$DAC0 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC0
3043
3044          ;*      MOV    $DAC0,$BDDAT ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
3045 012722 023737 001124 001126    CMP    $GDDAT,$BDDAT ;OK?
3046 012730 001403          BEQ    6$
3047
3048 012732 104014          ERROR 14          ;AA-11K DAC0 FAILED TO SET PATTERN 5252.
3049 012734 000137 013442    JMP    19$
3050
3051 012740          6$:
3052
3053          ;*      MOV    $GDDAT,$DAC1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC1
3054

```

```

3055 ;* MOV @DAC1,$BDDAT ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
3056 012760 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;OK
3057 012766 001403 ;* BEQ 7$ ;YES-
3058
3059 012770 104014 ;* ERROR 14 ;AA-11K DAC1 FAILED PATTERN 5252
3060 012772 000137 013442 ;* JMP 19$
3061
3062 012776 ;* 7$:
3063
3064 ;* MOV $GDDAT,@DAC2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC2
3065 ;*
3066 ;* MOV @DAC2,$BDDAT ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
3067 013016 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;OK?
3068 013024 001403 ;* BEQ 8$
3069
3070 013026 104014 ;* ERROR 14 ;AA-11K DAC2 FAILED PATTERN 5252
3071 013030 000137 013442 ;* JMP 19$
3072
3073 013034 ;* 8$:
3074
3075 ;* MOV $GDDAT,@DAC3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC3
3076 ;*
3077 ;* MOV @DAC3,$BDDAT ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
3078 013054 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;PATTERN OK?
3079 013062 001402 ;* BEQ 9$
3080
3081 013064 104014 ;* ERROR 14 ;AA-11K DAC3 FAILED PATTERN 5252
3082 013066 000565 ;* BR TST16 ;:
3083
3084 013070 012737 002525 001124 ;* 9$: MOV #2525,$GDDAT
3085
3086 ;* MOV $GDDAT,@DAC0 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC0
3087 ;*
3088 ;* MOV @DAC0,$BDDAT ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
3089 013116 023737 001124 001126 ;* CMP $GDDAT,$BDDAT ;OK?
3090 013124 001402 ;* BEQ 10$
3091
3092 013126 104014 ;* ERROR 14 ;AA-11K DAC0 FAILED TO SET PATTERN 2525
3093 013130 000544 ;* BR TST16 ;:
3094
3095 013132 ;* 10$:
3096
3097 ;* MOV $GDDAT,@DAC1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC1
3098 ;*
3099 ;* MOV @DAC1,$BDDAT ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
3100 013152 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;OK?
3101 013160 001402 ;* BEQ 11$ ;YES-
3102
3103 013162 104014 ;* ERROR 14 ;AA-11K DAC1 FAILED PATTERN 2525
3104 013164 000526 ;* BR TST16 ;:
3105
3106 013166 ;* 11$:
3107
3108 ;* MOV $GDDAT,@DAC2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC2

```

```

3109
3110
3111 013206 023737 001126 001124 ;* MOV @DAC2,$BDDAT ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
3112 013214 001402 CMP $BDDAT,$GDDAT ;OK?
3113 BEQ 12$
3114 013216 104014 ERROR 14 ;AA-11K DAC2 FAILED PATTERN 2525.
3115 013220 000510 BR TST16 ;;
3116
3117 013222 12$:
3118
3119 ;* MOV $GDDAT,@DAC3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC3
3120
3121 ;* MOV @DAC3,$BDDAT ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
3122 013242 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;PATTERN OK?
3123 013250 001402 BEQ 13$
3124
3125 013252 104014 ERROR 14 ;AA-11K DAC3 FAILED PATTERN.
3126 BR TST16 ;2525
3127 013254 000472 BR TST16 ;;
3128
3129 013256 012737 000000 001124 13$: MOV #0,$GDDAT
3130
3131 ;* MOV $GDDAT,@DAC0 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC0
3132
3133 ;* MOV @DAC0,$BDDAT ;/READ DEVICE REG DAC0,PUT DATA IN $BDDAT.
3134 013304 023737 001124 001126 ;* CMP $GDDAT,$BDDAT ;OK?
3135 013312 001402 BEQ 14$
3136
3137 013314 104014 ERROR 14 ;AA-11K DAC0 FAILED TO SET PATTERN 0
3138 013316 000451 BR TST16 ;;
3139
3140 013320 14$:
3141
3142 ;* MOV $GDDAT,@DAC1 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC1
3143
3144 ;* MOV @DAC1,$BDDAT ;/READ DEVICE REG DAC1,PUT DATA IN $BDDAT.
3145 013340 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;OK?
3146 013346 001402 BEQ 15$ ;YES-
3147
3148 013350 104014 ERROR 14 ;AA-11K DAC1 FAILED PATTERN 0
3149 013352 000433 BR TST16 ;;
3150
3151 013354 15$:
3152
3153 ;* MOV $GDDAT,@DAC2 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC2
3154
3155 ;* MOV @DAC2,$BDDAT ;/READ DEVICE REG DAC2,PUT DATA IN $BDDAT.
3156 013374 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;OK?
3157 013402 001402 BEQ 16$
3158
3159 013404 104014 ERROR 14 ;AA-11K DAC2 FAILED PATTERN 0
3160 013406 000415 BR TST16 ;;
3161
3162 013410 16$:

```



```

3163
3164 ;* MOV $GDDAT,2DAC3 ;/ PUT DATA FROM $GDDAT TO DEVICE REG DAC3
3165
3166 ;* MOV 2DAC3,$BDDAT ;/READ DEVICE REG DAC3,PUT DATA IN $BDDAT.
3167 013430 023737 001126 001124 ;* CMP $BDDAT,$GDDAT ;PATTERN OK?
3168
3169 013436 001401 BEQ TST16 ;:
3170 013440 104014 ERROR 14 ;:AA-11K DAC3 FAILED PATTERN 0.
3171
3172
3173
3174 013442 19$:
3175 ;:*****
3176 ;:TEST 16 I/O DEVICE TEST-AR11 OPTION
3177 ;:*****
3178 013442 000004 †ST16: SCOPE
3179
3180 013444 032737 002000 001562 BIT #BIT10,SR1 ;IS THIS DEVICE SELECTED FOR TEST?
3181 ;NOTE: DEFAULT=YES.
3182
3183 013452 001002 BNE 10$
3184 013454 000137 014012 JMP 7$
3185 013460 10$:
3186 013460 005037 001126 CLR $BDDAT
3187 013464 005037 001124 CLR $GDDAT
3188
3189 ;* MOV $BDDAT,2ARADS ;/ PUT DATA FROM $BDDAT TO DEVICE REG ARADS
3190 ;OK-WHAT WE JUST DID IS ASKED THE
3191 ;SLAVE MICRO-CODE TO WRITE THE CONTENTS
3192 ;OF THE AR11 CSR TO ZEROES.
3193 ;IF THE AR11 FAILS TO RESPOND TO ITS
3194 ;ADDRESS-THE SLAVE WILL SEND US AN
3195 ;ERROR CODE THAT CAUSES US TO SET $AERR=1
3196 ;IN THE SUPPORT ROUTINES.
3197
3198 013500 005737 037532 TST $AERR ;DEVICE PRESENT?
3199 013504 001403 BEQ 11$ ;YES-CONTINUE
3200 013506 104015 ERROR 15 ;AR11 FAILED TO RESPOND
3201 013510 000137 014012 JMP 7$ ;TO ADDR.
3202
3203 11$:
3204
3205 013524 042737 000200 001126 ;* MOV 2ARADS,$BDDAT ;/READ DEVICE REG ARADS,PUT DATA IN $BDDAT.
3206 013532 005037 001124 BIC #BIT7,$BDDAT ;CLEAR OUT FLAG
3207 013536 005737 001126 CLR $GDDAT
3208 013542 001402 TST $BDDAT ;IS CSR ZERO?
3209 BEQ 1$
3210 013544 104015 ERROR 15 ;FAILED TO ZERO AR11 A/D CSR
3211 013546 000521 BR TST17 ;:
3212 013550 012737 025100 001124 1$: MOV #BIT13!BIT11!BIT9!BIT6,$GDDAT ;NOW TRY A NEW BIT PATTERN
3213
3214
3215 ;* MOV $GDDAT,2ARADS ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARADS
3216

```

```

3217      ;*      MOV      2ARADS,$BDDAT      ;/READ DEVICE REG ARADS,PUT DATA IN $BDDAT.
3218 013576 042737 000200 001126      BIC      #BIT7,$BDDAT      ;IGNORE FLAG
3219 013604 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;BIT SET OK?
3220 013612 001402      BEQ      2$
3221
3222 013614 104015      ERROR   15      ;FAILED TO WRITE AR11 A/D CSR CORRECTLY
3223 013616 000475      BR      TST17      ;;
3224
3225 013620 012737 002420 001124 2$:      MOV      #BIT10!BIT8!BIT4,$GDDAT ;NOW TRY A NEW BIT PATTERN.
3226
3227
3228      ;*      MOV      $GDDAT,2ARADS      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARADS
3229
3230      ;*
3231 013646 042737 000200 001126      ;*      MOV      2ARADS,$BDDAT      ;/READ DEVICE REG ARADS,PUT DATA IN $BDDAT.
3232 013654 023737 001124 001126      BIC      #BIT7,$BDDAT      ;IGNORE FLAG
3233 013662 001402      CMP      $GDDAT,$BDDAT      ;BITS SET OK?
3234      BEQ      3$
3235 013664 104015      ERROR   15      ;FAILED TO WRITE #'NY' AD11K CSR CORRECTLY
3236 013666 000451      BR      TST17      ;;
3237
3238 013670 005037 001124      3$:      CLR      $GDDAT      ;NOW CLEAR ALL BITS SET
3239
3240      ;*      MOV      $GDDAT,2ARADS      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARADS
3241
3242      ;*
3243 013714 042737 000200 001126      ;*      MOV      2ARADS,$BDDAT      ;/READ DEVICE REG ARADS,PUT DATA IN $BDDAT.
3244 013722 005737 001126      BIC      #BIT7,$BDDAT      ;IGNORE DONE FLAG.
3245 013726 001402      TST     $BDDAT      ;CSR CLEAR
3246      BEQ      4$
3247 013730 104015      ERROR   15      ;AR11 A/D CSR FAILED TO CLEAR.
3248 013732 000427      BR      TST17      ;;
3249
3250 013734      4$:
3251
3252      ;*      MOV      2ARADB,MYTEMP      ;/READ DEVICE REG ARADB,PUT DATA IN MYTEMP.
3253 013744 012737 000001 001124      MOV      #BIT0,$GDDAT      ;NOW WE'LL SET THE A/D START BIT.
3254
3255      ;*      MOV      $GDDAT,2ARADS      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARADS
3256 013762 012737 000200 001124      MOV      #BIT7,$GDDAT      ;WHEN WE READ CSR BACK EXPECT
3257      ;START BIT TO CLEAR, DONE TO SET.
3258
3259      ;*      MOV      2ARADS,$BDDAT      ;/READ DEVICE REG ARADS,PUT DATA IN $BDDAT.
3260
3261 014000 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;START OK?
3262 014006 001401      BEQ      TST17      ;;
3263
3264 014010 104015      ERROR   15      ;CSR BAD AFTER A/D START
3265      ;BIT 0 SHOULD CLEAR ONLY A/D DONE SET.
3266 014012      7$:
3267
3268
3269      ;:*****
3270      ;*TEST 17      I/O DEVICE TEST-AR11 CLOCK OPTION

```

3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324

*
*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
*NOT CHECKED, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
*IT FROM "SRI" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SRI"
*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.

*
* IN THIS TEST WE WILL EXERCISE THE AR11 CLOCK. IF
* SRI IS UNALTERED, WE DO THIS TEST SINCE THE
* KWI1K IS A DEFAULT I/O DEVICE.
* PLEASE NOTE: THIS IS ONLY A BRIEF TEST OF THE AR11,
* TO SEE THAT IT ROUGHLY WORKS, FOR TESTING, YOU
* MUST RUN THE AR11 DIAGNOSTIC, COMPLETE

- *
* TESTS: 1. MAKE SURE AR11 RESPONDS TO ADDRESS
* (NOTE: IF THE AR11 ON YOUR I/O BUS USES A NON
* STANDARD ADDRESS, MODIFY "ARADS" WITH THE CORR
* 2. CLOCK CSR BITS 14,6,3, AND 1 SET
* 3. CLOCK CSR BITS 8 AND 2 SET
* 4. CLOCK CSR ZEROS.
* 5. START CLOCK A MODE 0, RATE 1MHZ,
* CHECK "ENABLE CNTR A" CLEARS,
* "A OVERFLOW FLAG" SETS
* 6. CLOCK A CSR ZEROS

*
* *****
* †ST17: SCOPE

```
014012 000004
014014 032737 002000 001562      BIT      #BIT10,SRI      ;IS THIS DEVICE SELECTED FOR TEST?
;NOTE: DEFAULT=YES.
014022 001002
014024 000137 014354      BNE      10$
014030      JMP      11$
10$:
014030 012737 040130 001124      MOV      #BIT14!BIT6!BIT4!BIT3,$GDDAT ;TEST THESE BITS
;*      MOV      $GDDAT,JARCS      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARCS
014046 005737 037532      TST      $AERR      ;DID AR11 RESPOND
014052 001403      BEQ      12$      ;TO ADDR? $AERR=0?
;IF YES-NEXT TEST.
```

```

3325 014054 104015          ERROR 15          ;AR11 CLOCK FAILED TO RESPOND TO ADDR.
3326 014056 000137 014354  JMP 11$
3327 014062          12$:
3328
3329 ;* MOV 2ARCS,$BDDAT ;/READ DEVICE REG ARCS,PUT DATA IN $BDDAT.
3330
3331 014072 023737 001124 001126  CMP $GDDAT,$BDDAT ;DID THESE BITS SET?
3332 014100 001403          BEQ 1$
3333
3334 014102 104015          ERROR 15          ;AR11 CLOCK R/W CSR ERROR.
3335 014104 000137 014354  JMP 11$
3336
3337 014110 012737 000424 001124 1$: MOV #BIT8!BIT4!BIT2,$GDDAT ;GET NEW PATTERN
3338
3339 ;* MOV $GDDAT,2ARCS ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARCS
3340
3341 ;* MOV 2ARCS,$BDDAT ;/READ DEVICE REG ARCS,PUT DATA IN $BDDAT.
3342
3343 014136 023737 001124 001126  CMP $GDDAT,$BDDAT ;DID THE BITS SET?
3344 014144 001403          BEQ 2$
3345
3346 014146 104015          ERROR 15          ;AR11 CLOCK R/W CSR ERROR.
3347 014150 000137 014354  JMP 11$
3348
3349 014154 005037 001124 2$: CLR $GDDAT ;WRITE ALL ZEROS.
3350
3351 ;* MOV $GDDAT,2ARCS ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARCS
3352
3353 ;* MOV 2ARCS,$BDDAT ;/READ DEVICE REG ARCS,PUT DATA IN $BDDAT.
3354 014200 012737 000020 001124  MOV #BIT4,$GDDAT ;BIT 4 ALWAYS SET
3355 014206 023737 001124 001126  CMP $GDDAT,$BDDAT
3356 014214 001403          BEQ 6$ ;YES-GOOD.
3357
3358 014216 104015          ERROR 15          ;AR11 CLOCK FAILED TO ZERO CSR.
3359 014220 000137 014354  JMP 11$
3360
3361 014224 012737 000003 001124 6$: MOV #BIT0!BIT1,$GDDAT ;PUT "ENABL CNTR A" AND RATE 1MHZ IN CSR
3362 014232 012737 177777 001126  MOV #177777,$BDDAT ;PRELOAD CNTR.
3363
3364 ;* MOV $BDDAT,2ARCB ;/ PUT DATA FROM $BDDAT TO DEVICE REG ARCB
3365
3366 ;* MOV $GDDAT,2ARCS ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARCS
3367
3368 ;* MOV 2ARCS,$BDDAT ;/READ DEVICE REG ARCS,PUT DATA IN $BDDAT.
3369 ;CLOCK SHOULD OVERFLOW, CLEAR
3370 014270 012737 000222 001124  MOV #BIT7!BIT4!BIT1,$GDDAT ;BIT 0 ("ENABLE CNTR A) SET
3371 ;BIT 7
3372 014276 023737 001124 001126  CMP $GDDAT,$BDDAT ;HAPPEN OK?
3373 014304 001401          BEQ 7$
3374
3375 014306 104015          ERROR 15          ;CLOCK CSR PROBLEM
3376
3377 014310 005037 001124 7$: CLR $GDDAT ;TRY CLEARING CSR
3378

```

```

3379          :*      MOV      $GDDAT,ARCS      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARCS
3380
3381          :*      MOV      ARCS,$BDDAT      ;/READ DEVICE REG ARCS,PUT DATA IN $BDDAT.
3382 014334 012737 000020 001124      MOV      #BIT4,$GDDAT      ;BIT 4 ALWAYS SET
3383 014342 023737 001124 001126      CMP      $GDDAT,$BDDAT
3384 014350 001401      BEQ      TST20      ;;
3385
3386 014352 104015      ERROR    15      ;CSR FAILED TO cLEAR
3387 014354
3388
3389
3390          ;*****
3391          ;*TEST 20      *TEST THE AR11 DISPLAY OPTION, IF "SR1" BIT 10=1 (SET)
3392          ;*
3393          ;*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
3394          ;*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
3395          ;*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
3396          ;*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
3397          ;*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
3398          ;*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
3399          ;*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
3400          ;*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
3401          ;*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
3402          ;*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
3403          ;*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
3404          ;*
3405          ;*IN THIS TEST WE WILL CHECK OUT THE AR11 DISPLAY OPTION,
3406          ;*IF BIT 10 OF "SR1" IS SET.
3407          ;*
3408          ;*****
3409          ;*ST20: SCOPE
3410
3411 014354 000004
3412 014356 032737 002000 001562      BIT      #BIT10,SR1      ;IS AR-11 SELECTED
3413 014364 001002      BNE      20$      ;IF =0, NO, SKIP THIS TEST.
3414 014366 000137 015132      JMP      19$
3415 014372 005037 001124      20$:    CLR      $GDDAT
3416
3417          ;*      MOV      $GDDAT,ARADS      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARDS
3418 014406 005737 037532      TST     $AERR      ;DID WE SUCCESSFULLY WRITE THE AR-11?
3419 014412 001403      BEQ      1$      ;YES IF $AERR=0
3420
3421 014414 104015      ERROR    15      ;FAILED TO ADDRESS AR-11. "SR1" BIT 3=1
3422          ;*      ;SELECTED THIS TEST. AR-11 ADDRESS IS
3423          ;*      ;IN LOC "ARADS"
3424 014416 000137 015132      JMP      19$
3425
3426 014422      1$:
3427
3428          ;*      MOV      ARADS,$BDDAT      ;/READ DEVICE REG ARDS,PUT DATA IN $BDDAT.
3429 014432 042737 000200 001126      BIC     #BIT7,$BDDAT      ;CLEAR FLAG
3430 014440 001403      BEQ      2$      ;DID CSR CLEAR?
3431
3432 014442 104015      ERROR    15      ;CLEAR CSR FAILED (AR-11)

```

```

3433 014444 000137 015132          JMP      19$
3434
3435 014450 012737 005104 001124 2$:  MOV     #5104,$GDDAT      ;TRY WRITING THESE BITS
3436                                ;*      MOV     $GDDAT,$ARDS    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARDS
3437                                ;*      MOV     $ARDS,$BDDAT   ;/READ DEVICE REG ARDS,PUT DATA IN $BDDAT.
3438                                ;      READ THEM BACK
3439                                ;      GET RID OF FLAG
3440                                ;      DID PATTERN W/R OK?
3441 014476 042737 000200 001126          BIC     #BIT7,$BDDAT
3442 014504 023737 001124 001126          CMP     $GDDAT,$BDDAT
3443 014512 001403                      BEQ     3$                ;YES-NEXT TEST.
3444
3445 014514 104015                      ERROR   15                ;AR11 D/A PATTERN 5024 FAILED TO W/R PROPERLY.
3446 014516 000137 015132          JMP     19$
3447
3448 014522 012737 002010 001124 3$:  MOV     #2010,$GDDAT    ;TRY NEW PATTERN.
3449                                ;*      MOV     $GDDAT,$ARDS    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARDS
3450                                ;*      MOV     $ARDS,$BDDAT   ;/READ DEVICE REG ARDS,PUT DATA IN $BDDAT.
3451                                ;      GET RID OF FLAG
3452 014550 042737 000200 001126          BIC     #BIT7,$BDDAT
3453 014556 023737 001124 001126          CMP     $GDDAT,$BDDAT
3454 014564 001403                      BEQ     4$                ;YES-NEXT TEST.
3455
3456 014566 104015                      ERROR   15                ;AR-11 DISPLAY REG PATTERN 2010 FAILED TO W/R PROPERLY.
3457 014570 000137 015132          JMP     19$
3458
3459 014574 005037 001124          4$:  CLR     $GDDAT          ;TRY TO CLEAR CSR.
3460                                ;*      MOV     $GDDAT,$ARDS    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARDS
3461                                ;*      MOV     $ARDS,$BDDAT   ;/READ DEVICE REG ARDS,PUT DATA IN $BDDAT.
3462                                ;      CLEAR FLAG BIT.
3463 014620 042737 000200 001126          BIC     #BIT7,$BDDAT
3464 014626 001403                      BEQ     5$
3465
3466 014630 104015                      ERROR   15                ;AR-11 DISPLAY REG. CSR FAILED TO CLEAR.
3467 014632 000137 015132          JMP     19$
3468
3469 014636 012737 001252 001124 5$:  MOV     #1252,$GDDAT
3470                                ;*      MOV     $GDDAT,$ARXB   ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARXB
3471                                ;*      MOV     $ARXB,$BDDAT   ;/READ DEVICE REG ARXB,PUT DATA IN $BDDAT.
3472                                ;      OK?
3473 014664 023737 001124 001126          CMP     $GDDAT,$BDDAT
3474 014672 001403                      BEQ     6$
3475
3476 014674 104015                      ERROR   15                ;AR-11 DACX FAILED TO SET PATTERN 1252.
3477 014676 000137 015132          JMP     19$
3478
3479 014702          6$:
3480                                ;*      MOV     $GDDAT,$ARYB   ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARYB
3481                                ;*      MOV     $ARYB,$BDDAT   ;/READ DEVICE REG ARYB,PUT DATA IN $BDDAT.
3482
3483
3484
3485
3486

```

```

3487 014722 023737 001126 001124      CMP      $BDDAT,$GDJAT      ;OK
3488 014730 001403      BEQ      7$                ;YES-
3489
3490 014732 104015      ERROR   15                ;AR-11 DACY FAILED PATTERN 1252
3491 014734 000137 015132      JMP      19$
3492
3493 014740 012737 000525 001124 7$:      MOV      #0525,$GDDAT
3494
3495      ;*      MOV      $GDDAT,$ARXB      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARXB
3496
3497      ;*      MOV      $ARXB,$BDDAT      ;/READ DEVICE REG ARXB,PUT DATA IN $BDDAT.
3498 014766 023737 001124 001126      CMP      $GDDAT,$BDDAT
3499 014774 001402      BEQ      10$              ;OK?
3500
3501 014776 104015      ERROR   15                ;AR-11 DACX FAILED TO SET PATTERN 0525
3502 015000 000454      BR      TST21             ;;
3503 015002
3504      10$:
3505      ;*      MOV      $GDDAT,$ARYB      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARYB
3506
3507      ;*      MOV      $ARYB,$BDDAT      ;/READ DEVICE REG ARYB,PUT DATA IN $BDDAT.
3508 015022 023737 001126 001124      CMP      $BDDAT,$GDDAT
3509 015030 001402      BEQ      13$              ;OK?
3510
3511 015032 104015      ERROR   15                ;AR-11 DACY FAILED PATTERN 0525
3512 015034 000436      BR      TST21             ;;
3513
3514 015036 012737 000000 001124 13$:     MOV      #0,$GDDAT
3515
3516      ;*      MOV      $GDDAT,$ARXB      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARXB
3517
3518      ;*      MOV      $ARXB,$BDDAT      ;/READ DEVICE REG ARXB,PUT DATA IN $BDDAT.
3519 015064 023737 001124 001126      CMP      $GDDAT,$BDDAT
3520 015072 001402      BEQ      14$              ;OK?
3521
3522 015074 104015      ERROR   15                ;AR-11 DACX FAILED TO SET PATTERN 0
3523 015076 000415      BR      TST21             ;;
3524
3525 015100      14$:
3526
3527      ;*      MOV      $GDDAT,$ARYB      ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARYB
3528
3529      ;*      MOV      $ARYB,$BDDAT      ;/READ DEVICE REG ARYB,PUT DATA IN $BDDAT.
3530 015120 023737 001126 001124      CMP      $BDDAT,$GDDAT
3531 015126 001401      BEQ      19$              ;OK?
3532
3533 015130 104015      ERROR   15                ;AR-11 DACY FAILED PATTERN 0
3534 015132      19$:

```

```

;*****
; *TEST 21 *TEST THAT THE AR11 CAN DISPLAY A SQUARE, SELECTED BY BIT 10 OF SR1,SR2
; *IN THIS TEST WE'LL DISPLAY A SQUARE ON THE
; *DISPLAY SCOPE VIA THE AR-11.

```

```

3535
3536
3537
3538
3539
3540

```

```

3541 ;*IF YOU HAVE AN AR11 AND SCOPE YOU MUST SELECT THIS
3542 ;*TEST BY SETTING THE APPROPRIATE BITS IN "SR1" (INDICATING YOU HAVE
3543 ;*AN AR-11) AND "SR2" (INDICATING YOU HAVE A SCOPE).
3544 ;*****
3545 015132 000004 †ST21: SCOPE
3546
3547 015134 032737 002000 001562 BIT #BIT10,SR1 ;/AR-11 SELECTED?
3548 015142 001475 BEQ TST22 ;
3549 015144 032737 010000 001564 BIT #BIT12,SR2 ;/SCOPE DISPLAY?
3550 015152 001471 BEQ TST22 ;
3551
3552 015154 005037 001124 CLR $GDDAT
3553 015160 005037 001126 CLR $BDDAT
3554 015164 012737 000001 001754 MOV #1,MYTEMP
3555 015172
3556 1$:
3557 ;* MOV $GDDAT,ARXB ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARXB
3558
3559 ;* MOV MYTEMP,ARDS ;/ PUT DATA FROM MYTEMP TO DEVICE REG ARDS
3560 015212 005237 001124 INC $GDDAT
3561 015216 032737 002000 001124 BIT #BIT10,$GDDAT
3562 015224 001762 BEQ 1$
3563 015226 2$:
3564
3565 ;* MOV $BDDAT,ARYB ;/ PUT DATA FROM $BDDAT TO DEVICE REG ARYB
3566
3567 ;* MOV MYTEMP,ARDS ;/ PUT DATA FROM MYTEMP TO DEVICE REG ARDS
3568 015246 005237 001126 INC $BDDAT
3569 015252 032737 002000 001126 BIT #BIT10,$BDDAT
3570 015260 001762 BEQ 2$
3571 015262 3$:
3572
3573 ;* MOV $GDDAT,ARXB ;/ PUT DATA FROM $GDDAT TO DEVICE REG ARXB
3574
3575 ;* MOV MYTEMP,ARDS ;/ PUT DATA FROM MYTEMP TO DEVICE REG ARDS
3576 015302 005337 001124 DEC $GDDAT
3577 015306 001365 BNE 3$
3578 015310 4$:
3579
3580 ;* MOV $BDDAT,ARYB ;/ PUT DATA FROM $BDDAT TO DEVICE REG ARYB
3581
3582 ;* MOV MYTEMP,ARDS ;/ PUT DATA FROM MYTEMP TO DEVICE REG ARDS
3583 015330 005337 001126 DEC $BDDAT
3584 015334 001365 BNE 4$
3585
3586
3587 ;*****
3588 ;*TEST 22 I/O DEVICE TEST LPS-11 A/D OPTION
3589 ;*****
3590 015336 000004 †ST22: SCOPE
3591
3592 015340 032737 010000 001562 BIT #BIT12,SR1 ;IS THIS DEVICE SELECTED FOR TEST?
3593 ;NOTE: DEFAULT=YES.
3594

```



```

3595 015346 001002          BNE 10$
3596 015350 000137 016014  JMP 7$
3597 015354          10$:
3598 015354 005037 001126  CLR $BDDAT
3599
3600          ;*      MOV $BDDAT, @LPADSR ; / PUT DATA FROM $BDDAT TO DEVICE REG LPADSR
3601          ; OK-WHAT WE JUST DID IS ASKED THE
3602          ; SLAVE MICRO-CODE TO WRITE THE CONTENTS
3603          ; OF THE A/D 'S CSR TO ZEROES.
3604          ; IF THE A/D FAILS TO RESPOND TO ITS
3605          ; ADDRESS-THE SLAVE WILL SEND US AN
3606          ; ERROR CODE THAT CAUSES US TO SET $AERR=1
3607          ; IN THE SUPPORT ROUTINES.
3608
3609 015370 005737 037532      TST $AERR ; DEVICE PRESENT?
3610 015374 001403          BEQ 11$ ; YES-CONTINUE
3611 015376 104016          ERROR 16 ; A/D FAILED TO RESPOND
3612 015400 000137 016014  JMP 7$ ; TO ADDR.
3613
3614 015404          11$:
3615
3616          ;*      MOV @LPADSR, $BDDAT ; / READ DEVICE REG LPADSR, PUT DATA IN $BDDAT.
3617 015414 042737 100200 001126  BIC #BIT15!BIT7, $BDDAT ; CLEAR OUT FLAGS
3618 015422 005037 001124  CLR $GDDAT
3619 015426 005737 001126  TST $BDDAT ; IS CSR ZERO?
3620 015432 001402          BEQ 1$
3621
3622 015434 104016          ERROR 16 ; FAILED TO ZERO A/D CSR
3623 015436 000572          BR TST23 ;
3624 015440 012737 025100 001124  1$: MOV #BIT13!BIT11!BIT9!BIT6, $GDDAT ; NOW TRY A NEW BIT PATTERN
3625
3626
3627          ;*      MOV $GDDAT, @LPADSR ; / PUT DATA FROM $GDDAT TO DEVICE REG LPADSR
3628
3629          ;*
3630 015466 042737 100200 001126  MOV @LPADSR, $BDDAT ; / READ DEVICE REG LPADSR, PUT DATA IN $BDDAT.
3631 015474 023737 001124 001126  BIC #BIT15!BIT7, $BDDAT ; IGNORE FLAGS
3632 015502 001402          CMP $GDDAT, $BDDAT ; BIT SET OK?
3633          BEQ 2$
3634 015504 104016          ERROR 16 ; FAILED TO WRITE A/D CSR CORRECTLY
3635 015506 000546          BR TST23 ;
3636
3637 015510 012737 012404 001124  2$: MOV #BIT12!BIT10!BIT8!BIT2, $GDDAT ; NOW TRY A NEW BIT PATTERN.
3638
3639
3640          ;*      MOV $GDDAT, @LPADSR ; / PUT DATA FROM $GDDAT TO DEVICE REG LPADSR
3641
3642          ;*
3643 015536 042737 100200 001126  MOV @LPADSR, $BDDAT ; / READ DEVICE REG LPADSR, PUT DATA IN $BDDAT.
3644 015544 023737 001124 001126  BIC #BIT15!BIT7, $BDDAT ; IGNORE FLAGS
3645 015552 001402          CMP $GDDAT, $BDDAT ; BITS SET OK?
3646          BEQ 3$
3647 015554 104016          ERROR 16 ; FAILED TO WRITE A/D CSR CORRECTLY
3648 015556 000522          BR TST23 ;

```

```

3649
3650 015560 005037 001124 3$: CLR $GDDAT ;NOW CLEAR ALL BITS SET
3651
3652 ;* MOV $GDDAT,ALPADSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPADSR
3653
3654 ;* MOV ALPADSR,$BDDAT ;/READ DEVICE REG LPADSR,PUT DATA IN $BDDAT.
3655 015604 042737 000200 001126 BIC #BI-7,$BDDAT ;IGNORE DONE FLAG.
3656 015612 005737 001126 TST $BDDAT ;CSR CLEAR?
3657 015616 001402 BEQ 4$
3658
3659 015620 104016 ERROR 16 ;A/D CSR FAILED TO CLEAR.
3660 015622 000500 BR TST23 ;;
3661
3662 015624 4$:
3663
3664 ;* MOV ALPADBR,MYTEMP ;/READ DEVICE REG LPADBR,PUT DATA IN MYTEMP.
3665 015634 012737 000001 001124 MOV #BIT0,$GDDAT ;NOW WE'LL SET THE A/D START BIT.
3666
3667 ;* MOV $GDDAT,ALPADSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPADSR
3668 015652 012737 000200 001124 MOV #BIT7,$GDDAT ;WHEN WE READ CSR BACK EXPECT
3669 ;START BIT TO CLEAR, DONE TO SET.
3670
3671 ;* MOV ALPADSR,$BDDAT ;/READ DEVICE REG LPADSR,PUT DATA IN $BDDAT.
3672
3673 015670 023737 001124 001126 CMP $GDDAT,$BDDAT ;START OK?
3674 015676 001402 BEQ 5$
3675
3676 015700 104016 ERROR 16 ;CSR BAD AFTER A/D START
3677 ;BIT 0 SHOULD CLEAR ONLY A/D DONE SET.
3678 015702 000450 BR TST23 ;;
3679
3680 015704 012737 000001 001124 5$: MOV #BIT0,$GDDAT ;OK-NOW WE'RE GONNA START ANOTHER
3681 ;A/D CONVERSION WITHOUT READING
3682 ;THE RESULTS OF THE LAST ONE. THIS
3683 ;SHOULD CAUSE THE ERROR FLAG
3684 ;AS WELL AS THE DONE FLAG TO SET.
3685
3686 ;* MOV $GDDAT,ALPADSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPADSR
3687
3688
3689 ;* MOV ALPADSR,$BDDAT ;/READ DEVICE REG LPADSR,PUT DATA IN $BDDAT.
3690 015732 012737 100200 001124 MOV #BIT15:BIT7,$GDDAT ;S/B
3691
3692 015740 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS 15 AND 7 SET?
3693
3694 015746 001401 BEQ 6$
3695
3696 015750 104016 ERROR 16 ;ERROR A/D BIT 15 AND 7 SHOULD BE
3697
3698 015752 005037 001124 6$: CLR $GDDAT ;MAKE SURE ERROR FLAG CLEARS
3699
3700 ;* MOV $GDDAT,ALPADSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPADSR
3701
3702 ;* MOV ALPADSR,$BDDAT ;/READ DEVICE REG LPADSR,PUT DATA IN $BDDAT.

```

3703	015776	042737	000200	001126	BIC	#BIT7,\$BDDAT	; IGNORE DON# FLAG.
3704	016004	005737	001126		TST	\$BDDAT	; IS CSR CLEAR?
3705	016010	001401			BEQ	7\$	
3706	016012	104016			ERROR	16	; A/D CSR FAILED TO CLEAR.
3707							
3708	016014			7\$:			
3709							
3710				;* MOV	2LPADBR,MYTEMP		; /READ DEVICE REG LPADBR,PUT DATA IN MYTEMP.
3711							

3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760

*TEST 23 I/O DEVICE TEST-LPS-CLOCK OPTION

*
*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE MB254 (IPBM), THE
*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
*IT FROM "SRI" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SRI"
*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.

*
* IN THIS TEST WE WILL EXERCISE THE LPS-11 CLOCK. IF
* SRI IS UNALTERED, WE DO THIS TEST SINCE THE
* LPS-11 CLOCK IS A DEFAULT I/O DEVICE.
* PLEASE NOTE: THIS IS ONLY A BRIEF TEST OF THE LPS-11 CLOC.
* TO SEE THAT IT ROUGHLY WORKS. FOR TESTING, YOU
* MUST RUN THE LPS-11 CLOCK DIAGNOSTIC. COMPLETE

- * TESTS:
- * 1. MAKE SURE LPS-11 CLOCK RESPONDS TO ADDRESS
* (NOTE: IF THE LPS-11 CLOCK ON YOU I/O BUS USES A NON
* STANDARD ADDRESS, MODIFY "LPADSR" WITH THE CORR
 - * 2. CLOCK CSR BITS 14,9,6,3, AND 1 SET
 - * 3. CLOCK CSR BITS 13,8 AND 2 SET
 - * 4. CLOCK CSR ZEROS.
 - * 5. START CLOCK A MODE D, RATE 1MHZ,
* CHECK "ENABLE CNTR A" CLEARS,
* "OVERFLOW FLAG" SETS
 - * 6. CLOCK A CSR ZEROS

*ST23: SCOPE

016024 000004

016026 032737 020000 001562

BIT #BIT13,SRI ;IS THIS DEVICE SELECTED FOR TEST?

3761
3762 016034 001002
3763 016036 000137 016346
3764 016042
3765

10\$:

BNE 10\$
JMP 11\$

:NOTE: DEFAULT=YES.

```

3766 016042 012737 041110 001124      MOV      #BIT14!BIT9!BIT6!BIT3,$GDDAT ;TEST THESE BITS
3767
3768
3769      ;*      MOV      $GDDAT,ALPCKSR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPCKSR
3770
3771 016060 005737 037532      TST      $AERR                      ;DID CLOCK RESPOND
3772 016064 001403      BEQ      12$                        ;TO ADDR? $AERR=0?
3773
3774 016066 104016      ERROR   16                          ;IF YES-NEXT TEST.
3775 016070 000137 016346      JMP      11$                        ;CLOCK FAILED TO RESPOND TO ADDR.
3776 016074
3777      12$:
3778
3779      ;*      MOV      ALPCKSR,$BDDAT  ;/READ DEVICE REG LPCKSR,PUT DATA IN $BDDAT.
3780
3781 016104 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;DID THESE BITS SET?
3782 016112 001403      BEQ      1$
3783
3784 016114 104016      ERROR   16                          ;CLOCK R/W CSR ERROR.
3785 016116 000137 016346      JMP      11$
3786
3787 016122 012737 020404 001124 1$:      MOV      #BIT13!BIT8!BIT2,$GDDAT ;GET NEW PATTERN
3788
3789      ;*      MOV      $GDDAT,ALPCKSR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPCKSR
3790
3791      ;*      MOV      ALPCKSR,$BDDAT  ;/READ DEVICE REG LPCKSR,PUT DATA IN $BDDAT.
3792
3793 016150 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;DID THE BITS SET?
3794 016156 001403      BEQ      2$
3795
3796 016160 104016      ERROR   16                          ;CLOCK R/W CSR ERROR.
3797 016162 000137 016346      JMP      11$
3798
3799 016166 005037 001124 2$:      CLR      $GDDAT                ;WRITE ALL ZEROS.
3800
3801      ;*      MOV      $GDDAT,ALPCKSR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPCKSR
3802
3803      ;*      MOV      ALPCKSR,$BDDAT  ;/READ DEVICE REG LPCKSR,PUT DATA IN $BDDAT.
3804 016212 005737 001126      TST      $BDDAT                ;DID CSR CLEAR?
3805 016216 001403      BEQ      3$                      ;YES-GOOD.
3806
3807 016220 104016      ERROR   16                          ;CLOCK FAILED TO ZERO CSR.
3808 016222 000137 016346      JMP      11$
3809
3810 016226 012737 000003 001124 3$:      MOV      #BIT0!BIT1,$GDDAT ;PUT "ENABL CNTR A" AND RATE 1MHZ IN CSR
3811 016234 012737 177777 001126      MOV      #177777,$BDDAT ;PRELOAD CNTR.
3812
3813      ;*      MOV      $BDDAT,ALPCKBR  ;/ PUT DATA FROM $BDDAT TO DEVICE REG LPCKBR
3814
3815      ;*      MOV      $GDDAT,ALPCKSR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPCKSR
3816
3817      ;*      MOV      ALPCKSR,$BDDAT  ;/READ DEVICE REG LPCKSR,PUT DATA IN $BDDAT.
3818      ;CLOCK SHOULD OVERFLOW, CLEAR
3819 016272 012737 000202 001124      MOV      #BIT7!BIT1,$GDDAT ;BIT 0 ("ENABLE CNTR A) SET

```

```

3820
3821 016300 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;BIT 7
3822 016306 001401                      BEQ      7$                ;HAPPEN OK?
3823
3824 016310 104016                      ERROR   16                ;CLOCK CSR PROBLEM
3825
3826 016312 005037 001124      7$:    CLR      $GDDAT      ;TRY CLEARING CSR
3827
3828                      ;*    MOV      $GDDAT,$LPCKSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPCKSR
3829
3830                      ;*    MOV      $LPCKSR,$BDDAT ;/READ DEVICE REG LPCKSR,PUT DATA IN $BDDAT.
3831 016336 005737 001126      TST      $BDDAT          ;DID CSR CLEAR?
3832 016342 001401                      BEQ      TST24            ;;
3833
3834 016344 104016                      ERROR   16                ;CSR FAILED TO CLEAR
3835 016346
3836
3837
3838                      ;*****
3839                      ;*TEST 24      *TEST THE LPS I/O, IF "SR1" BIT 15=1(SET)
3840                      ;*
3841                      ;*IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
3842                      ;*WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM), THE
3843                      ;*KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
3844                      ;*NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
3845                      ;*THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
3846                      ;*I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
3847                      ;*IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
3848                      ;*HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
3849                      ;*I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
3850                      ;*IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
3851                      ;*INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
3852                      ;*
3853                      ;*IN THIS TEST WE WILL CHECK OUT LPS-I/O IF BIT 15 OF SR1=1
3854                      ;*(SET). ALSO, DATA LOOP BACK (OUTPUT DATA BACK TO INPUT) WILL BE CHECKED
3855                      ;*IF "SR2" BIT 15 =1 (SET). FOR THIS, YOU MUST HAVE INSTALLED A LOOP-BACK
3856                      ;*CABLE.
3857                      ;*****
3857 016346 000004                      TST24:  SCOPE
3858 016350 032737 100000 001562      BIT      #BIT15,SR1
3859 016356 001002                      BNE     10$              ;1-TDRT-
3860 016360 000137 017044                      JMP     11$
3861 016364
3862 016364 005037 001124      10$:   CLR      $GDDAT      ;/SET TO XFERR ZEROS.
3863
3864                      ;*    MOV      $GDDAT,$LPIOSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOSR
3865 016400 005737 037532      TST      $AERR          ;=1 IF DEVICE NOT PRESENT,
3866 016404 001403                      BEQ     1$              ;IF 0, PRESENT, (CON'T).
3867
3868 016406 104016                      ERROR   16                ;DR11K #'DRN' DID NOT RESPOND WHEN
3869                      ;ADDRESSED. "SR1" BIT 'DRB' SELECTED
3870                      ;THIS OPTION. ADDRESS "DRCR" 'DRN'
3871                      ;CONTAINS ITS ADDRESS
3872
3873 016410 000137 017044                      JMP     11$

```

```

3874
3875 016414          1$:
3876
3877                ;*      MOV      @LPIOSR,$BDDAT  ;/READ DEVICE REG LPIOSR,PUT DATA IN $BDDAT.
3878 016424 005737 001126                ;*      TST      $BDDAT      ;DID CSR CLEAR?
3879 016430 001403                ;*      BEQ      2$
3880
3881 016432 104016                ERROR  16          ;LPS-I/O CSR FAILED TO CLEAR.
3882 016434 000137 017044                JMP      11$
3883
3884                2$:
3885                ;*      MOV      @LPIOOR,$GDDAT  ;/READ DEVICE REG LPIOOR,PUT DATA IN $GDDAT.
3886                ;*
3887                ;*      MOV      @LPIOIR,$GDDAT  ;/READ DEVICE REG LPIOIR,PUT DATA IN $GDDAT.
3888                ;*
3889 016460 012737 040100 001124                MOV      #BIT6!BIT14,$GDDAT ;LOAD WRITEABLE BITS.
3890
3891                ;*      MOV      $GDDAT,@LPIOSR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOSR
3892                ;*
3893                ;*      MOV      @LPIOSR,$BDDAT  ;/READ DEVICE REG LPIOSR,PUT DATA IN $BDDAT.
3894 016506 023737 001124 001126                ;*      CMP      $GDDAT,$BDDAT ;DID BITS SET?
3895 016514 001402                ;*      BEQ      3$
3896
3897 016516 104016                ERROR  16          ;LPS-I/O CSR BIT(S) FAILED.
3898 016520 000551                BR      TST25      ;;
3899
3900 016522 005037 001124                3$:      CLR      $GDDAT      ;TRY CLEARING CSR.
3901
3902                ;*      MOV      $GDDAT,@LPIOSR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOSR
3903                ;*
3904                ;*      MOV      @LPIOSR,$BDDAT  ;/READ DEVICE REG LPIOSR,PUT DATA IN $BDDAT.
3905 016546 005737 001126                ;*      TST      $BDDAT      ;DID CSR CLEAR?
3906 016552 001402                ;*      BEQ      4$          ;YES-NEXT TEST.
3907
3908 016554 104016                ERROR  16          ;LPS-I/O CSR FAILED TO CLEAR.
3909 016556 000532                BR      TST25      ;;
3910
3911 016560 032737 100000 001564                4$:      BIT      #BIT15,SR2    ;DOES THIS I/O HAVE A LOOP BACK
3912                ;*      BEQ      TST25          ;CABLE CONNECTED TO IT?
3913 016566 001526                ;*
3914                ;*      IF SR2 BIT15=1 THEN LOOP BACK.
3915
3916 016570 012737 052525 001124                MOV      #052525,$GDDAT ;SET FIRST PATTERN
3917
3918                ;*      MOV      $GDDAT,@LPIOOR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOOR
3919                ;*
3920                ;*      MOV      @LPIOSR,$BDDAT  ;/READ DEVICE REG LPIOSR,PUT DATA IN $BDDAT.
3921 016616 012737 000200 001124                ;*      MOV      #BIT7,$GDDAT ;EXPECT FLAGS TO SET.
3922 016624 023737 001126 001124                ;*      CMP      $BDDAT,$GDDAT ;DID THEY?
3923 016632 001402                ;*      BEQ      5$          ;YES-CON'T.
3924
3925 016634 104016                ERROR  16          ;LPS-I/O CSR FLAG(S) FAILED TO SET
3926                ;*      WHEN DATA XFERRD. IS THE OUTPUT
3927                ;*      REALLY CALLED BACK TO THE INPUT?

```



```

3928 016636 000502 BR TST25 ;;
3929
3930 016640 012737 052525 001124 5$: MOV #052525,$GDDAT ;PATTERN THAT SHOULD HAVE XFERRERD.
3931
3932 016656 023737 001126 001124 ;* MOV @LPIOIR,$BDDAT ;/READ DEVICE REG LPIOIR,PUT DATA IN $BDDAT.
3933 016664 001402 CMP $BDDAT,$GDLAT ;DATA SENT=DATA RECEIVED?
3934 BEQ 6$
3935
3936 016666 104016 ERROR 16 ;LPS-11 I/O DATA XFERR ERROR.
3937 016670 000465 BR TST25 ;;
3938
3939 016672 6$:
3940
3941 ;* MOV @LPIOSR,$BDDAT ;/READ DEVICE REG LPIOSR,PUT DATA IN $BDDAT.
3942 016702 005737 001126 TST $BDDAT ;DID "OUTPUT FLAG" SET.
3943 016706 100402 BMI 7$ ;SHOULD HAVE.
3944
3945 016710 104016 ERROR 16 ;LPS-11 I/O "OUTPUT FLAG" FAILED TO SET.
3946 016712 000454 BR TST25 ;;
3947
3948 016714 7$:
3949
3950 ;* MOV $GDDAT,@LPIOOR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOOR
3951 016724 012737 125252 001124 MOV #125252,$GDDAT ;NEW PATTERN
3952
3953 ;* MOV $GDDAT,@LPIOOR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOOR
3954
3955 ;* MOV @LPIOIR,$BDDAT ;/READ DEVICE REG LPIOIR,PUT DATA IN $BDDAT.
3956 016752 023737 001126 001124 CMP $BDDAT,$GDDAT ;PATTERN XFERR OK?
3957 016760 001402 BEQ 8$
3958
3959 016762 104016 ERROR 16 ;LPS I/O DATA XFERR ERROR
3960 016764 000427 BR TST25 ;;
3961
3962 016766 8$:
3963
3964 ;* MOV $GDDAT,@LPIOIR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOIR
3965 016776 005037 001124 CLR $GDDAT ;NOW XFERR ZERO PATTERN.
3966
3967 ;* MOV $GDDAT,@LPIOOR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOOR
3968
3969 ;* MOV @LPIOIR,$BDDAT ;/READ DEVICE REG LPIOIR,PUT DATA IN $BDDAT.
3970 017022 005737 001126 TST $BDDAT ;DID ZERO PATTERN GET XFERRERD?
3971 017026 001402 BEQ 9$
3972
3973 017030 104016 ERROR 16 ;LPS I/O DATA XFERR ERROR
3974 017032 000404 BR TST25 ;;
3975
3976 017034 9$:
3977
3978 ;* MOV $GDDAT,@LPIOSR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPIOSR
3979 017044 11$:
3980
3981

```

```

3982 .....*****
3983 *TEST 25 *TEST THE LPS D/A OPTION, IF "SR1" BIT 14=1 (SET)
3984 *
3985 *IN THIS TEST WE ARE CHECKING OUT AN I/O DEVICE ON THE LPA-11'S I/O BUS.
3986 *WE HAVE A CERTAIN AMOUNT OF CONFIDENCE THAT THE M8254 (IPBM) THE
3987 *KMC-11 AND THE DMC-11 WORK. IN THIS TEST, THE ARBITRATION CIRCUITRY IS
3988 *NOT CHECK, NOR THE DEVICE'S INTERRUPT CAPABILITIES.
3989 *THE DIAGNOSTIC IS SET UP FOR CERTAIN DEFAULT DEVICES ON THE
3990 *I/O BUS. IF THE DEFAULT DEVICE IS NOT ON THE I/O BUS, DELETE
3991 *IT FROM "SR1" (NOTE: AT THAT LOCATION, WE LIST CODES, AND EXPLAIN
3992 *HOW TO ADD AND DELETE OPTIONS). IF YOU HAVE A DEVICE ON THE
3993 *I/O BUS THAT IS NOT A DEFAULT DEVICE, AD IT TO "SR1"
3994 *IF A FAILURE OCCURS IN THIS TEST, YOU SHOULD RUN THE
3995 *INDIVIDUAL DIAGNOSTIC THAT EXERCISES THIS OPTION.
3996 *
3997 *IN THIS TEST WE WILL CHECK OUT THE LPS D/A OPTION.
3998 *IF BIT 14 OF "SR1" IS SET.
3999 *
4000 .....*****
4001 017044 000004 TS*25: SCOPE
4002
4003 017046 032737 040000 001562 BIT #BIT14,SR1 ;IS LPS D/A SELECTED?
4004 017054 001002 BNE 20$ ;IF =0, NO, SKIP THIS TEST.
4005 017056 000137 017622 JMP 19$
4006 017062 20$: CLR $GDDAT
4007 017062 005037 001124
4008
4009 ;* MOV $GDDAT, @LPDASR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDASR
4010 017076 005737 037532 TST $AERR ;DID WE SUCCESSFULLY WRITE THE LPS D/A?
4011 017102 001403 BEQ 1$ ;YES IF $AERR=0
4012
4013 017104 104016 ERROR 16 ;FAILED TO ADDRESS LPS D/A. "SR1" BIT 14=1
4014 ;SELECTED THIS TEST. LPS D/A ADDRESS IS
4015 ;IN LOC "LPDASR"
4016 017106 000137 017622 JMP 19$
4017
4018 017112 1$:
4019
4020 ;* MOV @LPDASR, $BDDAT ;/READ DEVICE REG LPDASR,PUT DATA IN $BDDAT.
4021 017122 042737 000200 001126 BIC #BIT7, $BDDAT ;CLEAR FLAG
4022 017130 001403 BEQ 2$ ;DID CSR CLEAR?
4023
4024 017132 104014 ERROR 14 ;CLEAR CSR FAILED (LPS-D/A)
4025 017134 000137 017622 JMP 19$
4026
4027 017140 012737 005124 001124 2$: MOV #5124, $GDDAT ;TRY WRITING THESE BITS
4028
4029 ;* MOV $GDDAT, @LPDASR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDASR
4030
4031 ;* MOV @LPDASR, $BDDAT ;/READ DEVICE REG LPDASR,PUT DATA IN $BDDAT.
4032 ;READ THEM BACK.
4033 017166 042737 000200 001126 BIC #BIT7, $BDDAT ;GET RID OF FLAG
4034 017174 023737 001124 001126 CMP $GDDAT, $BDDAT ;DID PATTERN
4035 017202 001403 BEQ 3$ ;

```

```

4036
4037 017204 104016          ERROR 16          ;LPS D/A PATTERN 5124 FAILED TO W/R PROPERLY.
4038 017206 000137 017622  JMP     19$
4039
4040 017212 012737 002012 001124 3$:  MOV     #2012,$GDDAT ;TRY NEW PATTERN.
4041
4042          ;*      MOV     $GDDAT,$LPDASR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDASR
4043
4044          ;*      MOV     $LPDASR,$BDDAT ;/READ DEVICE REG LPDASR,PUT DATA IN $BDDAT.
4045 017240 042737 000200 001126  BIC     #BIT7,$BDDAT ;GET RID OF FLAG.
4046 017246 023737 001124 001126  CMP     $GDDAT,$BDDAT ;DID PATTERN XFERR OK?
4047 017254 001403          BEQ     4$          ;YES-NEXT TEST.
4048
4049 017256 104016          ERROR 16          ;LPS D/A PATTERN 2012 FAILED TO W/R PROPERLY.
4050 017260 000137 017622  JMP     19$
4051
4052 017264 005037 001124          4$:  CLR     $GDDAT      ;TRY TO CLEAR CSR.
4053
4054          ;*      MOV     $GDDAT,$LPDASR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDASR
4055
4056          ;*      MOV     $LPDASR,$BDDAT ;/READ DEVICE REG LPDASR,PUT DATA IN $BDDAT.
4057 017310 042737 000200 001126  BIC     #BIT7,$BDDAT ;CLEAR FLAG BIT.
4058 017316 001403          BEQ     5$
4059
4060 017320 104016          ERROR 16          ;LPS D/A CSR FAILED TO CLEAR.
4061 017322 000137 017622  JMP     19$
4062
4063 017326 012737 005252 001124 5$:  MOV     #5252,$GDDAT
4064
4065          ;*      MOV     $GDDAT,$LPDAXR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDAXR
4066
4067          ;*      MOV     $LPDAXR,$BDDAT ;/READ DEVICE REG LPDAXR,PUT DATA IN $BDDAT.
4068 017354 023737 001124 001126  CMP     $GDDAT,$BDDAT ;OK?
4069 017362 001403          BEQ     6$
4070
4071 017364 104016          ERROR 16          ;D/A DACX FAILED TO SET PATTERN 5252.
4072 017366 000137 017622  JMP     19$
4073
4074 017372          6$:
4075
4076          ;*      MOV     $GDDAT,$LPDAYR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDAYR
4077
4078          ;*      MOV     $LPDAYR,$BDDAT ;/READ DEVICE REG LPDAYR,PUT DATA IN $BDDAT.
4079 017412 023737 001126 001124  CMP     $BDDAT,$GDDAT ;OK
4080 017420 001403          BEQ     7$          ;YES-
4081
4082 017422 104016          ERROR 16          ;DACY FAILED PATTERN 5252
4083 017424 000137 017622  JMP     19$
4084
4085 017430          7$:
4086 017430 012737 002525 001124  MOV     #2525,$GDDAT
4087
4088          ;*      MOV     $GDDAT,$LPDAXR ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDAXR
4089

```

G10

MAINDEC -11- DRLPA-A
DRLPA.P11 T25

MACY11 27(654) 13-DEC-77 12:58 PAGE 93
*TEST THE LPS D/A OPTION, IF "SR1" BIT 14=1 (SET)

SEQ 0123

```

4090                                ;*      MOV      2LPDAXR,$BDDAT  ;/READ DEVICE REG LPDAXR,PUT DATA IN $BDDAT.
4091 017456 023737 001124 001126  ;*      CMP      $GDDAT,$BDDAT
4092 017464 001402                    ;*      BEQ      10$          ;OK?
4093                                ;*
4094 017466 104016                    ERROR   16          ;D/A DACY FAILED TO SET PATTEN 2525
4095 017470 000454                    BR      TST26      ;;
4096                                ;*
4097 017472                    10$:
4098                                ;*
4099                                ;*      MOV      $GDDAT,2LPDAYR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDAYR
4100                                ;*
4101                                ;*      MOV      2LPDAYR,$BDDAT  ;/READ DEVICE REG LPDAYR,PUT DATA IN $BDDAT.
4102 017512 023737 001126 001124  ;*      CMP      $BDDAT,$GDDAT  ;OK?
4103 017520 001402                    ;*      BEQ      11$          ;YES-
4104                                ;*
4105 017522 104016                    ERROR   16          ;D/A DACY FAILED PATTERN 2525
4106 017524 000436                    BR      TST26      ;;
4107                                ;*
4108 017526                    11$:
4109                                ;*
4110 017526 012737 000000 001124 13$: MOV      #0,$GDDAT
4111                                ;*
4112                                ;*      MOV      $GDDAT,2LPDAXR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDAXR
4113                                ;*
4114                                ;*      MOV      2LPDAXR,$BDDAT  ;/READ DEVICE REG LPDAXR,PUT DATA IN $BDDAT.
4115 017554 023737 001124 001126  ;*      CMP      $GDDAT,$BDDAT  ;OK?
4116 017562 001402                    ;*      BEQ      14$          ;OK?
4117                                ;*
4118 017564 104016                    ERROR   16          ;D/A DACX FAILED TO SET PATTERN 0
4119 017566 000415                    BR      TST26      ;;
4120                                ;*
4121 017570                    14$:
4122                                ;*
4123                                ;*      MOV      $GDDAT,2LPDAYR  ;/ PUT DATA FROM $GDDAT TO DEVICE REG LPDAYR
4124                                ;*
4125                                ;*      MOV      2LPDAYR,$BDDAT  ;/READ DEVICE REG LPDAYR,PUT DATA IN $BDDAT.
4126 017610 023737 001126 001124  ;*      CMP      $BDDAT,$GDDAT  ;OK?
4127 017616 001401                    ;*      BEQ      TST26      ;;
4128                                ;*
4129 017620 104016                    ERROR   16          ;D/A DACY FAILED PATTERN 0
4130 017622                    19$:
4131                                ;*
4132 017622                    15$:
4133                                ;*
4134                                ;*
4135                                ;*
4136                                ;* *****
4137                                ;* *TEST 26 *TEST THAT THE LPS D/A CAN DISPLAY A SQUARE
4138                                ;* *IN THIS TEST WE'LL DISPLAY A SQUARE ON A
4139                                ;* *DISPLAY SCOPE VIA THE LPS-D/A
4140                                ;* *IF YOU HAVE AN LPS-D/A AND SCOPE YOU MUST SELECT THIS
4141                                ;* *TEST BY SETTING THE APPROPRIATE BITS IN "SR1" (INDICATING YOU HAVE
4142                                ;* *AN LPS D/A) AND "SR2" (INDICATING YOU HAVE A SCOPE).
4143 017622 000004                    ;* *****
                                ;* †ST26: SCOPE

```

```

4144
4145
4146 017624 032737 040000 001562 BIT #BIT14,SR1 ; -DART-
4147 017632 001475 BEQ TST27 ; /LPS D/A SELECTED?
4148 017634 032737 040000 001564 BIT #BIT14,SR2 ; /SCOPE DISPLAY?
4149 017642 001471 BEQ TST27 ;
4150
4151 017644 005037 001124 CLR $GDDAT
4152 017650 005037 001126 CLR $BDDAT
4153 017654 012737 000001 001754 MOV #1,MYTEMP
4154 017662
4155 1$:
4156 ;* MOV $GDDAT,ALPDAXR ; / PUT DATA FROM $GDDAT TO DEVICE REG LPDAXR
4157 ;*
4158 ;* MOV MYTEMP,ALPDASR ; / PUT DATA FROM MYTEMP TO DEVICE REG LPDASR
4159 017702 005237 001124 INC $GDDAT
4160 017706 032737 010000 001124 BIT #BIT12,$GDDAT
4161 017714 001762 BEQ 1$
4162 017716 2$:
4163 ;* MOV $BDDAT,ALPDAYR ; / PUT DATA FROM $BDDAT TO DEVICE REG LPDAYR
4164 ;*
4165 ;* MOV MYTEMP,ALPDASR ; / PUT DATA FROM MYTEMP TO DEVICE REG LPDASR
4166 017736 005237 001126 INC $BDDAT
4167 017742 032737 010000 001126 BIT #BIT12,$BDDAT
4168 017750 001762 BEQ 2$
4169 017752 3$:
4170 ;* MOV $GDDAT,ALPDAXR ; / PUT DATA FROM $GDDAT TO DEVICE REG LPDAXR
4171 ;*
4172 ;* MOV MYTEMP,ALPDASR ; / PUT DATA FROM MYTEMP TO DEVICE REG LPDASR
4173 017772 005337 001124 DEC $GDDAT
4174 017776 001365 BNE 3$
4175 020000 4$:
4176 ;* MOV $BDDAT,ALPDAYR ; / PUT DATA FROM $BDDAT TO DEVICE REG LPDAYR
4177 ;*
4178 ;* MOV MYTEMP,ALPDASR ; / PUT DATA FROM MYTEMP TO DEVICE REG LPDASR
4179 020020 005337 001126 DEC $BDDAT
4180 020024 001365 BNE 4$
4181
4182
4183
4184
4185
4186
4187
4188 ;*****
4189 ;*TEST 2: *TEST THAT THE AA11K # CAN DISPLAY A SQUARE, SELECTED BIT 3 IN SR1,SR2
4190 ;*IN THIS TEST WE'LL DISPLAY A SQUARE ON THE
4191 ;*DISPLAY SCOPE VIA THE AA-11K.
4192 ;*IF YOU HAVE AN AA11-K AND SCOPE YOU MUST SELECT THIS
4193 ;*TEST SETTING THE APPROPRIATE BITS IN "SR1" (INDICATING YOU HAVE
4194 ;*AN AA-11K) AND "SR2" (INDICATING YOU HAVE A SCOPE).
4195 ;*****
4196 020026 000004 ST27: SCOPE
4197 020030 012737 000001 001160 MOV #1,$TIMES ; ;DO 1 ITERATION

```

```

4198
4199
4200 020036 032737 000010 001562 BIT #BIT3,SP1 ; -DAAT-
4201 020044 001475 BEQ TST30 ; /AA-11K SELECTED?
4202 020046 032737 000010 001564 BIT #BIT3,SR2 ; /SCOPE DISPLAY?
4203 020054 001471 BEQ TST30 ;
4204
4205 020056 005037 001124 CLR $GDDAT
4206 020062 005037 001126 CLR $BDDAT
4207 020066 012737 000001 001754 MOV #1,MYTEMP
4208 020074 1$:
4209
4210 ;* MOV $GDDAT, @DAC0 ; / PUT DATA FROM $GDDAT TO DEVICE REG DAC0
4211 ;*
4212 ;* MOV MYTEMP, @AACSR ; / PUT DATA FROM MYTEMP TO DEVICE REG AACSR
4213 020114 005237 001124 INC $GDDAT
4214 020120 032737 010000 001124 BIT #BIT12, $GDDAT
4215 020126 001762 BEQ 1$
4216 020130 2$:
4217
4218 ;* MOV $BDDAT, @DAC1 ; / PUT DATA FROM $BDDAT TO DEVICE REG DAC1
4219 ;*
4220 ;* MOV MYTEMP, @AACSR ; / PUT DATA FROM MYTEMP TO DEVICE REG AACSR
4221 020150 005237 001126 INC $BDDAT
4222 020154 032737 010000 001126 BIT #BIT12, $BDDAT
4223 020162 001762 BEQ 2$
4224 020164 3$:
4225
4226 ;* MOV $GDDAT, @DAC0 ; / PUT DATA FROM $GDDAT TO DEVICE REG DAC0
4227 ;*
4228 ;* MOV MYTEMP, @AACSR ; / PUT DATA FROM MYTEMP TO DEVICE REG AACSR
4229 020204 005337 001124 DEC $GDDAT
4230 020210 001365 BNE 3$
4231 020212 4$:
4232
4233 ;* MOV $BDDAT, @DAC1 ; / PUT DATA FROM $BDDAT TO DEVICE REG DAC1
4234 ;*
4235 ;* MOV MYTEMP, @AACSR ; / PUT DATA FROM MYTEMP TO DEVICE REG AACSR
4236 020232 005337 001126 DEC $BDDAT
4237 020236 001365 BNE 4$
4238 ;
4239 ; DAAT 6,2
4240 ; *****
4241 ; *TEST 30 *TEST THAT USER MICRO-CODE CAN BE STARTED
4242 ; *IN THIS TEST, WE ARE GOING TO MAKE SURE
4243 ; *THAT RUN SETS, READY IN SETS, READY OUT CLEARS.
4244 ; *CALL INITIAL CONDITIONS
4245 ; *****
4246 020240 000004 †TST30: SCOPE
4247
4248 020242 004737 040504 JSR PC, $RESET ; ISSUE LPA-11 RESET.
4249 020246 032737 000001 001202 BIT #1, $PASS ; ON ALTERNATE PASSES, WE ALTERNATE MICROCODE.
4250 020254 001013 BNE 10$ ;
4251 020256 032777 002000 160654 BIT #BIT10, @SWR ; RUN ONLY DEDICATED

```

```

4252 020264 001007          BNE      10$          ;CODE?
4253                                ;YES DO THAT LOAD.
4254 020266 004537 037534    13$:   JSR      RS,$LOAD
4255 020272 044120          MMAST
4256 020274 012737 000115 001772    MOV      #'M,$VERSN
4257 020302 000412          BR       11$
4258 020304 004537 037534    10$:   JSR      RS,$LOAD
4259 020310 050124          DMAST
4260 020312 012737 000104 001772    MOV      #'D,$VERSN
4261 020320 032777 001000 160612    BIT      #BIT9,$SWR      ;RUN ONLY MULITE-SER CODE?
4262 020326 001357          BNE      13$          ;YES-GO BACK AND RELOAD
4263 020330
4264 020330 113737 001512 001773    11$:   MOVB     VERSN,$VERSN+1
4265 020336 052777 040000 161122    BIS      #BIT14,$KMA0    ;SET INIT.
4266 020344 012701 000004          MOV      #4,R1
4267 020350 004737 040542    12$:   JSR      PC,$DELAY
4268 020354 005301          DEC      R1
4269 020356 001374          BNE      12$
4270 020360 012777 104000 161100    MOV      #BIT15:BIT11,$KMA0 ;BIT 15 OF THE KMC CSR SHOULD BE
4271                                ;SET AT THIS TIME.
4272 020366 012701 000062          MOV      #50.,R1
4273
4274 020372 032777 000200 161066    1$:   BIT      #BIT7,$LPCI    ;BIT 7 OF CONTROL IN REG. (READY IN)
4275 020400 001014          BNE      2$          ;SHOULD BE SET AT INIT.
4276 020402 004737 040542          JSR      PC,$DELAY    ;DELAY FOR WHILE.
4277 020406 005301          DEC      R1
4278 020410 001370          BNE      1$
4279 020412 012737 000200 001124    MOV      #BIT7,$GDDAT
4280 020420 017737 161042 001126    MOV      $LPCI,$BDDAT
4281 020426 104017          ERROR   17          ;LPA-11 ERROR - READY IN NOT SET AT INIT.
4282 020430 000424          BR       TST31      ;;
4283
4284 020432 032777 000200 161032    2$:   BIT      #BIT7,$LPCO    ;CHECK READY OUT, IT SHOULD
4285 020440 001407          BEQ      3$          ;BE CLEAR AT INIT.
4286 020442 017737 161024 001126    MOV      $LPCO,$BDDAT
4287 020450 005037 001124          CLR      $GDDAT
4288 020454 104017          ERROR   17          ;LPW-11 ERROR - READY OUT SET AT INIT.
4289 020456 000411          BR       TST31      ;;
4290
4291 020460 105777 161010    3$:   TSTB     $LPSO      ;CHECK THE STATUS OUT REGISTER
4292                                ;IT SHOULD BE CLEAR AT INIT
4293                                ;;<IF CLEAR, NEXT TEST>
4294 020464 001406          BEQ      TST31
4295 020466 117737 161002 001126    MOVB     $LPSO,$BDDAT
4296 020474 005077 160424          CLR      $GDDAT
4297 020500 104017          ERROR   17          ;LPA-11 ERROR STATUS OUT REGISTER NOT CLEAR AT INIT.
4298
4299      ;*****
4300      ;*TEST 31
4301      ;*TEST FOR ERROR CODE 306
4302      ;*IN THIS TEST, WE ARE GOING TO ISSUE A CLOCK START
4303      ;*COMMAND. THE FIRST THING THAT SHOULD TAKE PLACE
4304      ;*IS "READY OUT" SHOULD SET. NEXT WE SHOULD GET
4305      ;*ERROR #306 (NO INIT COMMAND) SINCE WE HADN'T
4306      ;*INITIALIZED FIRST. AFTER, WE'LL CLEAR "READY OUT"
4307      ;*AND MAKE SURE THE STATUS OUT REG. CLEARS.

```

```

4306
4307 020502 000004
4308 020504 012737 000001 001160
4309
4310 020512 012737 000322 044112
4311
4312

```

```

;*****
↑ST31: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
MOV #BIT1!BIT4!BIT6!BIT7,KWT ;SET START CLOCK IN CLOCK TABLE
;(REQUEST DESCRIPTOR ARRAY)

```



```

4313 020520 012700 044112      MOV      #KWT,RO      ;GET ADDR. OF ARRAY
4314 020524 012760 000000 000002      MOV      #0,2(0)     ;SET CLOCK STATUS.
4315 020532 005060 000004      CLR      4(0)        ;SET CLOCK PRESET
4316 020536 010077 160734      MOV      RO,ALPADL
4317 020542 152777 000001 160716      BISB    #1,ALPCI     ;SET GO.
4318 020550 004737 040542      JSR      PC,SDELAY
4319 020554 105777 160712      TSTB    ALPCO        ;DID "READY OUT" SET?
4320 020560 100410 15
4321 020562 017737 160704 001126      MOV      ALPCO,$BDDAT
4322 020570 012737 000200 001124      MOV      #200,$GDDAT
4323 020576 104017 17
4324 020600 000432      ERROR   17          ;"READY OUT" FAILED TO SET (LPA-11 ERROR)
4325      BR      TST32    ;;
4326 020602 005037 001126      1$:      CLR      $BDDAT
4327 020606 012737 000306 001124      MOV      #306,$GDDAT      ;/EXPECT 306.
4328 020614 117737 160654 001126      MOVB    ALPS0,$BDDAT     ;/SEE WHAT WE GET.
4329 020622 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;DID LPA-11 RETURN ERROR CODE
4330      ;#306 - "NO INITIALIZE COMMAND"?
4331 020630 001402      BEQ     2$
4332
4333 020632 104017 17
4334 020634 000414      ERROR   17          ;LPA-11 ERROR - ERROR CODE #306 NOT RETURNED
4335      BR      TST32    ;;
4336 020636 005077 160630      2$:      CLR      ALPCO        ;CLEAR "READY OUT"
4337      ;ERROR OR STATUS OUT REG. SHOULD CLEAR.
4338 020642 105777 160626      TSTB    ALPS0        ;DID IT CLEAR?
4339 020646 001402      BEQ     TST32       ;;<YES - NEXT TEST>
4340
4341 020650 012737 000000 001124      MOV      #0,$GDDAT      ;/EXPECT 0.
4342 020656 117737 160612 001126      MOVB    ALPS0,$BDDAT     ;/SEE WHAT WE GET.
4343 020664 104017 17
4344      ERROR   17          ;STATUS OUT REG. NOT CLEARED WHEN "READY OUT" CLEARED (L
4345      ;*****
4346      ;*TEST 32          *TEST THAT WE CAN GENERATE ERROR CODE 314
4347      ;*
4348      ;*IN THIS TEST WE WILL TRY TO GENERATE ERROR CODE 314
4349      ;*(ODD ADDRESS SPECIFIED) FOR REQUEST DESCRIPTOR ARRAY (ROA)
4350      ;*
4351
4352      ;*****
4353 020666 000004      †TST32:  SCOPE
4354 020670 012700 054130      MOV      #DEVLST,RO    ;GET AN ADDR.
4355 020674 005200      INC      RO            ;MAKE IT ODD.
4356 020676 010077 160574      MOV      RO,ALPADL     ;STORE ADDR. AS ADDR OF A
4357      ;REQUEST DESCRIPTOR ARRAY.
4358 020702 152777 000001 160556      BISB    #1,ALPCI     ;SET GO
4359 020710 005000      CLR      RO            ;TIME OUT COUNTER
4360 020712
4361 020712 105777 160554      1$:      TSTB    ALPCO        ;READY OUT SET?
4362 020716 100412      BMI     2$            ;YES-EXIT LOOP
4363 020720 105200      INCB    RO            ;NO-LOOP OVERFLOW OCCUR?
4364 020722 100373      BPL     1$           ;NO-LOOP.
4365 020724 012737 000200 001124      MOV      #200,$GDDAT   ;/EXPECT 200.
4366 020732 017737 160534 001126      MOV      ALPC0,$BDDAT   ;/SEE WHAT WE GET.

```

```

4367 020740 104017          ERROR 17          ;READY OUT NOT SET ON INIT CMND. (KMC-ERROR)
4368 020742 000414          BR      TST33          ;;
4369
4370 020744          2$:
4371 020744 012737 000314 001124      MOV      #314,$GDDAT          ;/EXPECT 314.
4372 020752 117737 160516 001126      MOVB    2LPS0,$BDDAT          ;/SEE WHAT WE GET.
4373 020760 122737 000314 001126      CMPB    #314,$BDDAT          ;DID CORRECT ERROR CODE GET RETURNED?
4374 020766 001402          BEQ      TST33          ;;
4375
4376 020770 104017          ERROR 17          ;ERROR CODE 314 NOT RETURNED FOR ODD ADDRESS FOR RDA
4377
4378 020772 000400          BR      TST33          ;;
4379
4380          ;*****
4381          ;*TEST 33          *TEST THAT THE DEVICE LIST CAN BE VERIFIED
4382          ;
4383          ;INIT TEST IN THIS TEST WE MAKE SURE THE KMC MICRO-
4384          ;CODE CAN FIND ALL ADDRESSES OF DEVICES TO BE TESTED.
4385          ;*****
4386          ;*****
4387 020774 000004          †TST33: SCOPE
4388
4389 020776 012700 000012      MOV      #10.,R0
4390 021002 012701 054130      MOV      #DEVLST,R1          ;CLEAR DEVICE LIST AREA
4391 021006 005021          CLR      (1)+
4392 02.010 005003          CLR      R3
4393 02.012 012721 000001      1$: MOV      #1,(1)+          ;PUT A 1 INTO ALL POSTIONS OF
4394          ;DEVICE LIST SO THAT LPA WON'T RETURN
4395          ;ARR ERROR UNLESS BAD ADDR.
4396          ;OTHERWISE IT WOULD RETURN ADDR. ERROR
4397          ;CODE ON ADDR. ZERO.
4398 021016 005300          DEC      R0
4399 021020 001374          BNE     1$
4400 021022 012701 054132      MOV      #DEVLST+2,R1
4401 021026 032737 000002 001562      BIT      #BIT1,SR1          ;KW11K CLOCK??
4402 021034 001403          BEQ     10$          ;NO
4403 021036 013721 001570      MOV      KW11K,(1)+          ;YES FIX ADDR.
4404 021042 000425          BR      20$
4405 021044 032737 002000 001562 10$: BIT      #BIT10,SR1          ;AR11 CLOCK??
4406 021052 001405          BEQ     11$          ;NO
4407 021054 013711 001610      MOV      AR11K,(1)          ;YES FIX AR11K CLOCK ADDR.
4408 021060 062721 000004          ADD     #4,(1)+
4409 021064 000414          BR      20$
4410 021066 032737 020000 001562 11$: BIT      #BIT13,SR1          ;LPS CLOCK?
4411 021074 001405          BEQ     12$          ;NO-ERROR!!!
4412 021076 013711 001612      MOV      LPS11,(1)          ;YES FIX LPS11 CLOCK ADDR.
4413 021102 062721 000004          ADD     #4,(1)+
4414 021106 000403          BR      20$
4415 021110          12$:
4416 021110 104017          ERROR 17          ;NO CLOCK SELECTED FOR TEST...
4417          ;CAN NOT DO ANY MORE TESTS WITHOUT
4418          ;CLOCK PRESENT!
4419          ;YOU MUST SELECT A CLOCK.
4420 021112 000137 030576      JMP     ETEST

```

4421											
4422	021116	012721	000001		20\$:	MOV	#1,(1)+				
4423	021122	032737	000001	001562		BIT	#BIT0,SP1			;AD11K FOR TEST??	
4424	021130	001403				BEQ	21\$				
4425	021132	013721	001566			MOV	AD11K,(1)+			;YES FIX AD11K ADDR.	
4426	021136	000420				BR	30\$				
4427	021140	032737	002000	001562	21\$:	BIT	#BIT10,SR1			;AR11 A/D?	
4428	021146	001403				BEQ	22\$				
4429	021150	013721	001610			MOV	AR11K,(1)+				
4430	021154	000411				BR	30\$				
4431	021156	032737	010000	001562	22\$:	BIT	#BIT12,SR1			;LPS A/D?	
4432	021164	001403				BEQ	23\$				
4433	021166	013721	001612			MOV	LPS11,(1)+				
4434	021172	000402				BR	30\$				
4435	021174	012721	000001		23\$:	MOV	#1,(1)+				
4436											
4437	021200	032737	000020	001562	30\$:	BIT	#BIT4,SR1			;2ND AD11K?	
4438	021206	001403				BEQ	31\$				
4439	021210	013721	001576			MOV	AD11K2,(1)+				
4440	021214	000402				BR	32\$				
4441	021216	012721	000001		31\$:	MOV	#1,(1)+				
4442											
4443	021222	032737	000010	001562	32\$:	BIT	#BIT3,SR1			;AR11K?	
4444	021230	001403				BEQ	33\$				
4445	021232	013721	001574			MOV	AR11K,(1)+				
4446	021236	000424				BR	40\$				
4447	021240	032737	002000	001562	33\$:	BIT	#BIT10,SR1			;AR11?	
4448	021246	001405				BEQ	34\$				
4449	021250	013711	001610			MOV	AR11K,(1)				
4450	021254	062721	000016			ADD	#16,(1)+				
4451	021260	000413				BR	40\$				
4452	021262	032737	040000	001562	34\$:	BIT	#BIT14,SR1			;LPS D/A?	
4453	021270	001405				BEQ	35\$				
4454	021272	013711	001612			MOV	LPS11,(1)				
4455	021276	062721	000016			ADD	#16,(1)+				
4456	021302	000402				BR	40\$				
4457	021304	012721	000001		35\$:	MOV	#1,(1)+				
4458											
4459	021310	032737	000004	001562	40\$:	BIT	#BIT2,SR1			;DR11K?	
4460	021316	001403				BEQ	41\$				
4461	021320	013721	001572			MOV	DR11K1,(1)+				
4462	021324	000413				BR	50\$				
4463	021326	032737	100000	001562	41\$:	BIT	#BIT15,SR1			;LPS I/O?	
4464	021334	001405				BEQ	42\$				
4465	021336	013711	001612			MOV	LPS11,(1)				
4466	021342	062721	000010			ADD	#10,(1)+			;FIX LPS11 I/O ADDR.	
4467	021346	000402				BR	50\$				
4468	021350	012721	000001		42\$:	MOV	#1,(1)+				
4469											
4470	021354	032737	000040	001562	50\$:	BIT	#BITS,SR1			;DR11K#2?	
4471	021362	001403				BEQ	51\$				
4472	021364	013721	001600			MOV	DR11K2,(1)+				
4473	021370	000402				BR	52\$				
4474	021372	012721	000001		51\$:	MOV	#1,(1)+				

```

4475 021376 032737 000200 001562 52$: BIT #BIT7,SR1 ;DR11K #3 ?
4476 021404 001403 BEQ 53$
4477 021406 013721 001602 MOV DR11K3,(1)+
4478 021412 000402 BR 54$
4479 021414 012721 000001 53$: MOV #1,(1)+
4480 021420 032737 000400 001562 54$: BIT #BIT8,SR1 ;DR11K #4 ?
4481 021426 001403 BEQ 55$
4482 021430 013721 001604 MOV DR11K4,(1)+
4483 021434 000402 BR 56$
4484 021436 012721 000001 55$: MOV #1,(1)+
4485 021442 032737 001000 001562 56$: BIT #BIT9,SR1 ;DR11K #5 ?
4486 021450 001403 BEQ 57$
4487 021452 013721 001606 MOV DR11K5,(1)+
4488 021456 000402 BR 60$
4489 021460 012711 000001 57$: MOV #1,(1)
4490 021464 60$:
4491
4492
4493 021464 113737 001512 054131 MOVB VERSN,DEVLST+1
4494
4495 021472 004737 040626 4$: JSR PC,SRESET ;RESET SYSTEM
4496 021476 012777 054130 157772 MOV #DEVLST,ALPADL ;SET INIT POINTER
4497 021504 052777 000001 157754 BIS #BIT0,ALPCI ;SET GO!
4498 021512 005000 CLR R0
4499 021514 004737 040542 5$: JSR PC,SDELAY ;DELAY
4500 021520 132777 000200 157740 BITB #BIT7,ALPCI ;DONE?
4501 021526 001010 BNE 6$ ;YES!
4502 021530 105200 INCB R0 ;NO-WAIT
4503 021532 001370 BNE 5$ ;BUT NOT TOO LONG...
4504 021534 005037 001124 CLR $GDDAT
4505 021540 005037 001126 CLR $BDDAT
4506
4507 021544 104017 ERROR 17 ;LPA FAILED TO FINISH INIT.
4508 021546 000423 BR TST34
4509 021550 105777 157716 6$: TSTB ALPC0 ;DID CONTROL OUT SET WITH ERROR??
4510 021554 100020 BPL TST34
4511
4512 021556 117737 157712 001126 MOVB ALPS0,$BDDAT ;GET STATUS
4513 021564 012737 000000 001124 MOV #0,$GDDAT
4514 021572 122737 000326 001126 CMPB #326,$BDDAT ;DEVICE ADDR ERROR?
4515 021600 001402 BEQ 7$
4516 021602 104017 ERROR 17 ;ERROR KNOWN CHECK RETURNED ERROR
4517 ;CODE AGAINST DOCUMENTATION.
4518 021604 000404 BR TST34
4519 021606 017737 157670 001124 7$: MOV ALPMS1,$GDDAT ;BAD DEVICE ADDRESS
4520 021614 104017 ERROR 17 ;DEVICE DOES NOT EXSISTS? SOMETHINGS
4521 ;FUNNY! PREVIOUS TESTS DIDN'T
4522 ;SHOW THIS ERROR-I'D SAY KMC-ERROR.
4523
4524
4525 ;*****
4526 ;*TEST 34 *TRY TO GENERATE ERROR CODE 312
4527 ;*
4528 ;*TEST THAT WE CAN GENERATE ERROR CODE 312
;*"USER NOT ACTIVE FOR STOP COMMAND"

```

```

4529
4530
4531 021616 000004
4532
4533 021620 012737 000403 055066      MOV      #403,JOBO      ;SET USER#1 STOP COMMAND IN MODE
4534
4535 021626 012737 055164 055072      MOV      #JOB0U,JOBO+4 ;WORD OF RDA #0
4536 021634 012777 055066 157634      MOV      #JOB0,ALPADL  ;SET UP USW ADDR.
                                ;LOAD ADDR OF RDA.

```

```

;*****i*
;ST34: SCOPE

```

```

4537 021642 152777 000001 157616      BISB  #1, @LPCI      ;SET GO.
4538 021650 005000      CLR   RO
4539 021652      15:      TSTB  @LPCO      ;READ OUT SET?
4540 021652 105777 157614      BMI   2$        ;YES-EXIT LOOP
4541 021656 100412      INCB  RO        ;NO-LOOP OVERFLOW OCCUR?
4542 021660 105200      BPL   1$        ;NO-LOOP
4543 021662 100373
4544
4545 021664 012737 000200 001124      MOV   #200, $GDDAT ;/EXPECT 200.
4546 021672 117737 157574 001126      MOVB  @LPC0, $BDDAT ;/SEE WHAT WE GET.
4547 021700 104017      ERROR 17        ;READY OUT NOT SET ON CMND (KMC ERROR)
4548 021702 000413      BR    TST35     ;;
4549
4550 021704      2$:      MOV   #312, $GDDAT ;/EXPECT 312.
4551 021704 012737 000312 001124      MOVB  @LPS0, $BDDAT ;/SEE WHAT WE GET.
4552 021712 117737 157556 001126      CMPB  #312, $BDDAT ;DID CORRECT ERROR CODE GET RETURNED
4553 021720 122737 000312 001126      BEQ   TST35     ;;
4554 021726 001401      ERROR 17        ;ERROR CODE 312 NOT RETURNED STOP USER ISSUED AND NO USE
4555 021730 104017
4556
4557
4558 ;*****
4559 ;*TEST 35      *TRY TO GENERATE ERROR CODE 310
4560 ;*
4561 ;*MAKE USRE WE CAN GENERATE ERROR CODE 310
4562 ;*"MULTIPLE INITIALIZE COMMAND".
4563 ;*
4564 ;*****
4565 021732 000004      †TST35: SCOPE
4566
4567 021734 004737 040626      JSR   PC, SRESET ;LPA-11 RESET
4568
4569 021740 012777 054130 157530      MOV   #DEVLST, @LPADL ;SET ADDR. OF DEVICE LIST IN RDA WORD
4570 021746 005037 054130      CLR   DEVLST    ;INDICATE SIZING
4571 021752 113737 001512 054131      MOVB  VERSN, DEVLST+1 ;SET VERSION NUMBER
4572
4573 021760 152777 000001 157500      BISB  #1, @LPCI      ;SET GO.
4574 021766 004737 040542      JSR   PC, SDLAY
4575
4576 021772      1$:      TSTB  @LPCI      ;READY IN SET?
4577 021776 105777 157470      BMI   2$        ;
4578 022000 012737 000200 001124      MOV   #200, $GDDAT ;/EXPECT 200.
4579 022006 017737 157454 001126      MOV   @LPC1, $BDDAT ;/SEE WHAT WE GET.
4580 022014 104017      ERROR 17        ;READY IN NOT SET ON INIT CMND
4581 022016 000456      BR    TST36     ;;
4582
4583 022020      2$:      ;NOTE-WE HAVE ALREADY DETERMINED
4584 ;DEVICES IN LIST ARE VALID
4585 ;SO WE SHOULD GET A "000"
4586 ;STATUS BACK FOR STAT REG.
4587 022020 012737 000000 001124      MOV   #0, $GDDAT  ;/EXPECT 0.
4588 022026 117737 157440 001126      MOVB  @LPC0, $BDDAT ;/SEE WHAT WE GET.
4589
4590 022034 122737 000000 001126      CMPB  #0, $BDDAT  ;DID AN ERROR CODE GET RETURNED?

```

```

4591 022042 001405          BEQ      3$          ;NO-NEXT
4592
4593 022044 117737 157424 001126  MOVB   @LPS0,$BDDAT ;PUT ERROR CODE IN FOR TYPEOUT.
4594 022052 104017          ERROR  17          ;KNOWN ERROR CODE RETURN ON INIT COMMAND
4595 022054 000437          BR      TST36       ;;
4596
4597 022056 012777 054130 157412 3$:  MOV    #DEVLST,@LPADL ;RESET ADDR.
4598 022064 152777 000001 157374  BISB   #1,@LPCI     ;SET GO-DO ANOTHER INITIALIZE CMND.
4599 022072 005000          CLR    RO
4600 022074 004737 040542          JSR PC,SDELAY
4601 022100 105777 157366          TSTB  @LPC0       ;READY OUT SET
4602 022104 100410          BMI   5$
4603 022106 012737 000200 001124  MOV    #200,$GDDAT ;/EXPECT 200.
4604 022114 017737 157352 001126  MOV    @LPC0,$BDDAT ;/SEE WHAT WE GET.
4605 022122 104017          ERROR  17          ;READY OUT NOT SET ON INIT COMMAND
4606 022124 000413          BR      TST36       ;;
4607
4608 022126          5$:
4609 022126 012737 000310 001124  MOV    #310,$GDDAT ;/EXPECT 310.
4610 022134 117737 157334 001126  MOVB   @LPS0,$BDDAT ;/SEE WHAT WE GET.
4611 022142 122737 000310 001126  CMPB   #310,$BDDAT ;NOW WE EXPECT ERROR CODE 310
4612 022150 001401          BEQ      TST36       ;;
4613
4614 022152 104017          ERROR  17          ;ERROR CODE 310 NOT RETURN ON DOUBLE INIT.
4615
4616          ;*****
4617          ;*TEST 36          *TEST INTERRUPTS
4618
4619          ;*
4620          ;*IN THIS TEST WE'LL MAKE SURE READY IN AND READY OUT
4621          ;*CAN GENERATE INTERRUPTS
4622          ;*
4623
4624          ;*****
4625 022154 000004          †TST36: SCOPE
4626
4627 022156 004737 040626          JSR    PC,SRESET
4628 022162 013700 001556          MOV    VECT1,RO    ;GET VECTOR ADDR.
4629 022166 012710 022350          MOV    #4$(0)     ;SET ADDR FOR READY OUT INTERRUPT.
4630 022172 012760 022270 000004  MOV    #1$4(0)    ;SET ADDR FOR READY IN INTERRUPT
4631 022200 012777 054130 157270  MOV    #DEVLST,@LPADL ;SET ADDR. OF DEVICE LIST IN RDA WORD
4632 022206 005037 054130          CLR    DEVLST     ;INDICATE SIZING
4633 022212 113737 001512 054131  MOVB   VERSN,DEVLST+1 ;SET VERSION NUMBER
4634
4635 022220 152777 000001 157240  BISB   #1,@LPCI     ;SET GO.
4636 022226 004737 040542          JSR PC,SDELAY
4637 022232 052777 000100 157226  BIS    #100,@LPCI
4638
4639 022240 004737 040542          JSR    PC,SDELAY ;INTERRUPT RIGHT AWAY.

```



```

4694 022456 000765 BR 1$ ;IF A CLOCK IS IN SYSTEM, THEN SELEC IT.
4695 ;IF YOU DESIRE TO RUN TIS DIAG. WITH NO CLOCK,
4696 ;DESELECT ANOLOG OPTIONS.
4697
4698 022460 012737 000001 044112 2$: MOV #BIT0,KWT ;SELECT MODE:START CLOCK
4699 022466 123727 001772 000115 CMPB $VERSN,#'M ;MULTI-REQUEST MODE?
4700 022474 001003 BNE 4$
4701 022476 052737 000010 044112 BIS #BIT3,KWT
4702 022504 4$:
4703 022504 012737 000503 044114 MOV #BIT1!BIT8!BIT0!BIT6,KWT+2 ;RATE:1MHZ,GO BIT, REPEATED INTERRUPT.IN
4704 022512 012737 177766 044116 MOV #-10,KWT+4 ;CLOCK INTERRUPT EVERY 10 USEC.
4705 022520 012777 044112 156750 MOV #KWT,$LPADL ;PUT ADDR. IN LPA ADDR. REG
4706 022526 052777 000001 156732 BIS #BIT0,$LPCI ;SET GO.
4707
4708 022534 004737 040542 JSR PC,$DELAY
4709 022540 032777 000200 156720 BIT #BIT7,$LPCI ;READY IN SET
4710 022546 001010 BNE 3$
4711 022550 012737 000200 001124 MOV #200,$GDDAT ;/EXPECT 200.
4712 022556 017737 156704 001126 MOV $LPCI,$BDDAT ;/SEE WHAT WE GET.
4713 022564 104017 ERROR 17 ;READY IN FAILED TO SET WHEN
4714 ;CLOCK STARTED.
4715 022566 000412 BR TST40 ;
4716
4717 022570 105777 156676 3$: TSTB $LPCO ;ERROR CODE RETURNED?
4718 022574 100007 BPL TST40 ;
4719 022576 012737 000000 001124 MOV #0,$GDDAT ;/EXPECT 0.
4720 022604 117737 156664 001126 MOVB $LPC0,$BDDAT ;/SEE WHAT WE GET.
4721
4722 022612 104017 ERROR 17 ;ERROR CODE RETURNED ON
4723 ;CLOCK START.
4724
4725
4726 ;*****
4727 ;*TEST 40 *TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DRI1K #1
4728 ;*
4729 ;*IN THIS TEST WE'LL MAKE SURE THAT DRI1K #1 WILL PASS DATA.
4730 ;*SR1 BIT2 SELECTS THIS DRI1K FOR TEST. SR2 BIT2 INDICTES
4731 ;*THAT THE INPUT IS CALLED BACK TO THE OUTPUT.
4732 ;*
4733 ;* NOTE: THIS TEST IS A MULTI-USER MICRO-CODE TEST ONLY!
4734 ;* IF DEDICATE USER MICRO-CODE, THIS TEST WILL BE BY-PASSED.
4735 ;*
4736 ;*THIS TEST WAS DESIGNED IN ORDER TO FIND A WILD LPA-11 SYSTEM FAULT.
4737 ;*I DON'T REALLY EXPECT AN ERROR HERE UNLESS YOU'VE GOT A HEAVY FAULT.
4738 ;*ARBITRATION LOGIC IS ONE BIG REASON WHY YOU COULD FAIL. ANOTHER
4739 ;*IS THAT INTERRUPT VECTOR COULD BE WRONG. ALSO THE DMC MICRO-CODE MIGHT
4740 ;*NOT BE EXECUTING PROPERLY.
4741 ;*
4742 ;*****
4743 022614 000004 †ST40: SCOPE
4744 022616 012737 000001 001160 MOV #1,$TIMES ;;DO 1 ITERATION
4745
4746
4747 022624 122737 000115 001772 CMPB #'M,$VERSN ;/-DRLT-
;RUNNING MULTI-USER MICRO-CODE?

```

```

4748 022632 001402 023340 BEQ 1$ ;YES-CONTINUE.
4749 022634 000137 023340 JMP 20$ ;NO-EXIT THIS TEST.
4750
4751 022640 012700 055066 1$: MOV #JOB0,R0 ;CLEAR OUT THE RDA TABLES
4752 022644 005020 055066 CLR (R0)+
4753 022646 020027 055556 CMP R0,#JOB4
4754 022652 001374 055556 BNE 25$
4755
4756
4757 022654 032737 000004 001562 BIT #BIT2,SR1 ;IS THIS DRI1K SELECTED FOR TEST?
4758 022662 001002 000004 BNE 2$ ;YES-TEST IT.
4759 022664 000137 023340 JMP 20$ ;NO-EXIT THIS TEST.
4760
4761 022670 012700 055322 2$: MOV #JOB2,R0 ;PICK UP THIS JOB ADDRESS.
4762 022674 012710 000012 MOV #BIT1:BIT3,(0) ;SET START AND MULTI-USER.
4763 022700 052710 000420 BIS #BIT4:BIT8,(0) ;SELECT DIGITAL INPUT SINGLE CHAN.
4764 022704 012760 000401 000002 3$: MOV #257,2(0) ;SET WORD COUNT
4765 022712 012760 055420 000004 MOV #JOB2U,4(0) ;USW ADDR.
4766 022720 105060 000006 CLR 6(0) ;NO EXT.
4767 022724 112760 000201 000007 MOV #201,7(0) ;MAKE TWO BUFFERS AVAIL.
4768 022732 012760 056246 000010 MOV #BUFF0,10(0) ;1ST BUFFER ADDR.
4769 022740 012760 057250 000014 MOV #BUFF1,14(0) ;2ND BUFFER ADDR.
4770 022746 005060 000012 CLR 12(0) ;NO EXT.
4771 022752 012760 000200 000054 MOV #200,54(0) ;DELAY =200 TICKS.
4772 022760 012760 000000 000056 MOV #1-1,56(0) ;SET CHAN #=DRI1K#-1.
4773 022766 012760 000001 000060 MOV #1,60(0) ;SAMPLING ONLY ONE CHANNEL.
4774 022774 012760 000001 000062 MOV #1,62(0) ;ON PULSE BETWEEN SAMPLE IS NO EXTERNAL TRIG.
4775 023002 005060 000064 CLR 64(0) ;NO START/EVENT MAKR WORD
4776 023006 005060 000066 CLR 66(0) ;NO START MASK
4777 023012 005060 000070 CLR 70(0) ;NO EVENT MASK.
4778
4779 023016 012701 055204 MOV #JOB1,R1 ;OK-NOW LETS SET UP THE OUTPUT.
4780 023022 012711 000012 MOV #BIT1:BIT3,(1) ;SET START AND MULTI USER.
4781 023026 052711 000620 BIS #BIT4:BIT7:BIT8,(1) ;SELECT DIGITAL OUT.
4782 023032 012761 000401 000002 MOV #257,2(1) ;WORD COUNT
4783 023040 012761 055302 000004 MOV #JOB1U,4(1) ;SET USW ADDR.
4784 023046 105061 000006 CLR 6(1) ;NO EXT.
4785 023052 112761 000202 000007 MOV #202,7(1)
4786 023060 012761 060252 000010 MOV #BUFF2,10(1) ;SET BUFFER #1 ADDR.
4787 023066 012761 061254 000014 MOV #BUFF3,14(1) ;SET BUFFER #2 ADDR.
4788 023074 012761 060252 000020 MOV #BUFF2,20(1) ;THE THIRD OUTPUT IS NECESSARY
4789 ;TO COMPENSATE FOR SAMPLES LOST
4790 023102 105061 000012 CLR 12(1) ;NO EXT.
4791 023106 012761 000200 000054 MOV #200,54(1) ;DELAY 1 TICK.
4792 023114 012761 000000 000056 MOV #1-1,56(1) ;SET CHAN #= DR#-1
4793 023122 012761 000001 000060 MOV #1,60(1) ;SAMPLE ONLY ONE CHAN.
4794 023130 012761 000001 000062 MOV #1,62(1) ;ONE TICK BETWEEN SAMPLE
4795 023136 005061 000064 CLR 64(1) ;NO START/EVENT
4796 023142 005061 000066 CLR 66(1) ;NO START MASK.
4797 023146 005061 000070 CLR 70(1) ;NO EVENT MASK.
4798
4799 023152 012700 125252 MOV #125252,R0 ;FILL OUTPUT BUFFER W/PATTERN
4800 023156 012701 001002 MOV #514,R1
4801 023162 012702 060252 MOV #BUFF2,R2

```

```

4802 023166 010022          4$:  MOV      RO,(2)+
4803 023170 005100          COM      RO
4804 023172 005301          DEC      R1
4805 023174 001374          BNE     4$
4806
4807 023176 012700 056246    MOV     #BUFF0,R0      ;CLEAR INPUT BUFFER
4808 023202 012701 001002    MOV     #514.,R1
4809 023206 005020          5$:  CLR     (0)+
4810 023210 005301          DEC     R1
4811 023212 001375          BNE     5$
4812
4813 023214 012737 000011 055066    MOV     #11,JOBO      ;OK,NOW WE GOTTO STOP THE CLOCK
4814 023222 012737 000000 055070    MOV     #0,JOBO+2     ;SO THAT IT DOESN'T START UNTIL
4815 023230 012737 000011 055440    MOV     #11,JOB3     ;DRIK PARAMETER SET UP, THEN AS
4816 023236 012737 000503 055442    MOV     #503,JOB3+2  ;JOB #4, WE START CLOCK.
4817 023244 012737 177000 055444    MOV     #177000,JOB3+4
4818 023252 012737 000004 001124    MOV     #4,$GDDAT    ;FOUR JOBS.
4819 023260 012737 000002 002014    MOV     #2,FUDGE     ;/SET UP A FUDGE FACTOR TO
4820
4821 023266 004737 036026    JSR    PC, FLASH    ;/COMPENSATE FOR THE TWO KW11K JOBS
4822
4823 023272 032737 000004 001564    BIT     #BIT2,SR2    ;/CABLED TOGETHER?
4824 02330C 001417          BEQ     TST41        ;
4825 023302 012700 001003    MOV     #515.,R0     ;/NOW CHECK DATA IF CABLED.
4826 023306 012701 060252    MOV     #BUFF2,R1
4827 023312 012702 056246    MOV     #BUFF0,R2
4828 023316 005300          6$:  DEC     RO
4829 023320 001407          BEQ     TST41        ;
4830 023322 022221          CMP     (R2)+,(R1)+  ;
4831 023324 001774          BEQ     6$
4832 023326 014137 001124    MOV     -(R1),$GDDAT
4833 023332 014237 001126    MOV     -(R2),$BDDAT
4834
4835 023336 104012          ERROR  12
4836 023340
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855

```

```

*****
*TEST 41      *TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DRI1K #2
*
*IN THIS TEST WE'LL MAKE SURE THAT DRI1K #2 WILL PASS DATA.
*SR1 BITS SELECTS THIS DRI1K FOR TEST. SR2 BITS INDICTES
*THAT THE INPUT IS CALLED BACK TO THE OUTPUT.
*
*      NOTE: THIS TEST IS A MULTI-USER MICRO-CODE TEST ONLY!
*      IF DEDICATE USER MICRO-CODE, THIS TEST WILL BE BY-PASSED.
*
*THIS TEST WAS DESIGNED IN ORDER TO FIND A WILD LPA-11 SYSTEM FAULT.
*I DON'T REALLY EXPECT AN ERROR HERE UNLESS YOU'VE GOT A HEAVY FAULT.
*ARBITRATION LOGIC IS ONE BIG REASON WHY YOU COULD FAIL. ANOTHER
*IS THAT INTERRUPT VECTOR COULD BE WRONG. ALSO THE DMC MICRO-CODE MIGHT
*NOT BE EXECUTING PROPERLY.
*
*****

```

```

4856 023340 000004          TST41: SCOPE
4857 023342 012737 000001 001160      MOV      #1,$TIMES      ;;DO 1 ITERATION
4858
4859
4860 023350 122737 000115 001772          CMPB     #'M,$VERSN    ;/-DRLT-
4861 023356 001402          BEQ      1$           ;:RUNNING MULTI-USER MICRO-CODE?
4862 023360 000137 024064          JMP      20$         ;:YES-CONTINUE.
4863
4864 023364 012700 055066          1$:      MOV      #JOB0,RO ;CLEAR OUT THE RDA TABLES
4865 023370 005020          25$:     CLR      (RO)+
4866 023372 020027 055556          CMP      RO,#JOB4
4867 023376 001374          BNE     25$
4868
4869
4870 023400 032737 000040 001562          BIT      #BITS,SR1    ;IS THIS DR11K SELECTED FOR TEST?
4871 023406 001002          BNE     2$           ;:YES-TEST IT.
4872 023410 000137 024064          JMP      20$         ;:NO-EXIT THIS TEST.
4873
4874 023414 012700 055322          2$:      MOV      #JOB2,RO ;PICK UP THIS JOB ADDRESS.
4875 023420 012710 000012          MOV      #BIT1!BIT3,(0) ;:SET START AND MULTI-USER.
4876 023424 052710 000420          BIS      #BIT4!BIT8,(0) ;:SELECT DIGITAL INPUT SINGLE CHAN.
4877 023430 012760 000401 000002          3$:      MOV      #257,2(0) ;:SET WORD COUNT
4878 023436 012760 055420 000004          MOV      #JOB2U,4(0) ;:USW ADDR.
4879 023444 105060 000006          CLRB    6(0)         ;:NO EXT.
4880 023450 112760 000201 000007          MOV     #201,7(0)    ;:MAKE TWO BUFFERS AVAIL.
4881 023456 012760 056246 000010          MOV     #BUFF0,10(0) ;:1ST BUFFER ADDR.
4882 023464 012760 057250 000014          MOV     #BUFF1,14(0) ;:2ND BUFFER ADDR.
4883 023472 005060 000012          CLR     12(0)        ;:NO EXT.
4884 023476 012760 000200 000054          MOV     #200,54(0)   ;:DELAY =200 TICKS.
4885 023504 012760 000001 000056          MOV     #2-1,56(0)  ;:SET CHAN #=DR11K#-1.
4886 023512 012760 000001 000060          MOV     #1,60(0)    ;:SAMPLING ONLY ONE CHANNEL.
4887 023520 012760 000001 000062          MOV     #1,62(0)    ;:ON PULSE BETWEEN SAMPLE IS NO EXTERNAL TRIG.
4888 023526 005060 000064          CLR     64(0)        ;:NO START/EVENT MAKR WORD
4889 023532 005060 000066          CLR     66(0)        ;:NO START MASK
4890 023536 005060 000070          CLR     70(0)        ;:NO EVENT MASK.
4891
4892 023542 012701 055204          MOV     #JOB1,R1    ;OK-NOW LETS SET UP THE OUTPUT.
4893 023546 012711 000012          MOV     #BIT1!BIT3,(1) ;:SET START AND MULTI USER.
4894 023552 052711 000620          BIS     #BIT4!BIT7!BIT8,(1) ;:SELECT DIGITAL OUT.
4895 023556 012761 000401 000002          MOV     #257,2(1)   ;:WORD COUNT
4896 023564 012761 055302 000004          MOV     #JOB1U,4(1) ;:SET USW ADDR.
4897 023572 105061 000006          CLRB    6(1)         ;:NO EXT.
4898 023576 112761 000202 000007          MOV     #202,7(1)   ;:SET BUFFER #1 ADDR.
4899 023604 012761 060252 000010          MOV     #BUFF2,10(1) ;:SET BUFFER #2 ADDR.
4900 023612 012761 061254 000014          MOV     #BUFF3,14(1) ;:THE THIRD OUTPUT IS NECESSARY
4901 023620 012761 060252 000020          MOV     #BUFF2,20(1) ;:TO COMPENSATE FOR SAMPLES LOST
4902
4903 023626 105061 000012          CLRB    12(1)        ;:NO EXT.
4904 023632 012761 000200 000054          MOV     #200,54(1)  ;:DELAY 1 TICK.
4905 023640 012761 000001 000056          MOV     #2-1,56(1)  ;:SET CHAN #= DR#-1
4906 023646 012761 000001 000060          MOV     #1,60(1)    ;:SAMPLE ONLY ONE CHAN.
4907 023654 012761 000001 000062          MOV     #1,62(1)    ;:ONE TICK BETWEEN SAMPLE
4908 023662 005061 000064          CLR     64(1)        ;:NO START/EVENT
4909 023666 005061 000066          CLR     66(1)        ;:NO START MASK.

```

K11

MAINDEC -11- DRLPA-A
DRLPA.P11 T41

MACY11 27(654) 13-DEC-77 12:58 PAGE 110
*TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DRIIK #2

SEQ 0140

```

4910 023672 005061 000070 CLR 70(1) ;NO EVENT MASK.
4911
4912 023676 012700 125252 MOV #125252,R0 ;FILL OUTPUT BUFFER W/PATTERN
4913 023702 012701 001002 MOV #514,R1
4914 023706 012702 060252 MOV #BUFF2,R2
4915 023712 010022 4$: MOV RO,(2)+
4916 023714 005100 COM RO
4917 023716 005301 DEC R1
4918 023720 001374 BNE 4$
4919
4920 023722 012700 056246 MOV #BUFF0,RO ;CLEAR INPUT BUFFER
4921 023726 012701 001002 MOV #514,R1
4922 023732 005020 5$: CLR (0)+
4923 023734 005301 DEC R1
4924 023736 001375 BNE 5$
4925
4926 023740 012737 000011 055066 MOV #11,JOBO ;OK,NOW WE GOTTO STOP THE CLOCK
4927 023746 012737 000000 055070 MOV #0,JOBO+2 ;SO THAT IT DOESN'T START UNTIL
4928 023754 012737 000011 055440 MOV #11,JOB3 ;DRIIK PARAMETER SET UP,THEN AS
4929 023762 012737 000503 055442 MOV #503,JOB3+2 ;JOB #4,WE START CLOCK.
4930 023770 012737 177000 055444 MOV #177000,JOB3+4
4931 023776 012737 000004 001124 MOV #4,$GDDAT ;FOUR JOBS.
4932 024004 012737 000002 002014 MOV #2,$FUDGE ;/SET UP A FUDGE FACTOR TO
4933 ;/COMPENSATE FOR THE TWO KWIK JOBS
4934 024012 004737 036026 JSR PC, FLASH ;/GO DO THEM!
4935
4936 024016 032737 000040 001564 BIT #BITS,SR2 ;/CABLED TOGETHER?
4937 024024 001417 BEQ TST42 ;
4938 024026 012700 001003 MOV #515,RO ;/NOW CHECK DATA IF CABLED.
4939 024032 012701 060252 MOV #BUFF2,R1
4940 024036 012702 056246 MOV #BUFF0,R2
4941 024042 005300 6$: DEC RO
4942 024044 001407 BEQ TST42 ;
4943 024046 022221 CMP (R2)+,(R1)+ ;
4944 024050 001774 BEQ 6$
4945 024052 014137 001124 MOV -(R1),$GDDAT
4946 024056 014237 001126 MOV -(R2),$BDDAT
4947
4948 024062 104012 ERROR 12
4949 024064 20$:

```

```

4950
4951
4952
4953 *****
4954 *TEST 42 *TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DRIIK #3
4955 *
4956 *IN THIS TEST WE'LL MAKE SURE THAT DRIIK #3 WILL PASS DATA.
4957 *SR1 BIT7 SELECTS THIS DRIIK FOR TEST. SR2 BIT7 INDICTES
4958 *THAT THE INPUT IS CALLED BACK TO THE OUTPUT.
4959 *
4960 * NOTE: THIS TEST IS A MULTI-USER MICRO-CODE TEST ONLY!
4961 * IF DEDICATE USER MICRO-CODE, THIS TEST WILL BE BY-PASSED.
4962 *
4963 *THIS TEST WAS DESIGNED IN ORDER TO FIND A WILD LPA-11 SYSTEM FAULT.
4964 *I DON'T REALLY EXPECT AN ERROR HERE UNLESS YOU'VE GOT A HEAVY FAULT.

```

; *ARBITRATION LOGIC IS ONE BIG REASON WHY YOU COULD FAIL. ANOTHER
; *IS THAT INTERRUPT VECTOR COULD BE WRONG. ALSO THE DMC MICRO-CODE MIGHT
; *NOT BE EXECUTING PROPERLY.
; *

†ST42: SCOPE

4964									
4965									
4966									
4967									
4968									
4969	024064	000004							
4970	024066	012737	000001	001160		MOV	#1,\$TIMES		; ;DO 1 ITERATION
4971									
4972									
4973	024074	122737	000115	001772		CMPB	#'M,\$VERSN		; /-DRLT-
4974	024102	001402				BEQ	1\$; RUNNING MULTI-USER MICRO-CODE?
4975	024104	000137	024610			JMP	20\$; YES-CONTINUE.
4976									; NO-EXIT THIS TEST.
4977	024110	012700	055066		1\$:	MOV	#JOB0,RO		; CLEAR OUT THE RDA TABLES
4978	024114	005020			25\$:	CLR	(RO)+		
4979	024116	020027	055556			CMP	RO,#JOB4		
4980	024122	001374				BNE	25\$		
4981									
4982									
4983	024124	032737	000200	001562		BIT	#BIT7,SR1		; IS THIS DR11K SELECTED FOR TEST?
4984	024132	001022				BNE	2\$; YES-TEST IT.
4985	024134	000137	024610			JMP	20\$; NO-EXIT THIS TEST.
4986									
4987	024140	012700	055322		2\$:	MOV	#JOB2,RO		; PICK UP THIS JOB ADDRESS.
4988	024144	012710	000012			MOV	#BIT1!BIT3,(0)		; SET START AND MULTI-USER.
4989	024150	052710	000420			BIS	#BIT4!BIT8,(0)		; SELECT DIGITAL INPUT SINGLE CHAN.
4990	024154	012760	000401	000002	3\$:	MOV	#257,,2(0)		; SET WORD COUNT
4991	024162	012760	055420	000004		MOV	#JOB20,4(0)		; USW ADDR.
4992	024170	105060	000006			CLRB	6(0)		; NO EXT.
4993	024174	112760	000201	000007		MOVB	#201,7(0)		; MAKE TWO BUFFERS AVAIL.
4994	024202	012760	056246	000010		MOV	#BUFF0,10(0)		; 1ST BUFFER ADDR.
4995	024210	012760	057250	000014		MOV	#BUFF1,14(0)		; 2ND BUFFER ADDR.
4996	024216	005060	000012			CLR	12(0)		; NO EXT.
4997	024222	012760	000200	000054		MOV	#200,54(0)		; DELAY =200 TICKS.
4998	024230	012760	000002	000056		MOV	#3-1,56(0)		; SET CHAN #=DR11K#-1.
4999	024236	012760	000001	000060		MOV	#1,60(0)		; SAMPLING ONLY ONE CHANNEL.
5000	024244	012760	000001	000062		MOV	#1,62(0)		; ON PULSE BETWEEN SAMPLE IS NO EXTERNAL TRIG.
5001	024252	005060	000064			CLR	64(0)		; NO START/EVENT MAKR WORD
5002	024256	005060	000066			CLR	66(0)		; NO START MASK
5003	024262	005060	000070			CLR	70(0)		; NO EVENT MASK.
5004									
5005	024266	012701	055204			MOV	#JOB1,R1		; OK-NOW LETS SET UP THE OUTPUT.
5006	024272	012711	000012			MOV	#BIT1!BIT3,(1)		; SET START AND MULTI USER.
5007	024276	052711	000620			BIS	#BIT4!BIT7!BIT8,(1)		; SELECT DIGITAL OUT.
5008	024302	012761	000401	000002		MOV	#257,,2(1)		; WORD COUNT
5009	024310	012761	055302	000004		MOV	#JOB10,4(1)		; SET USW ADDR.
5010	024316	105061	000006			CLRB	6(1)		; NO EXT.
5011	024322	112761	000202	000007		MOVB	#202,7(1)		
5012	024330	012761	060252	000010		MOV	#BUFF2,10(1)		; SET BUFFER #1 ADDR.
5013	024336	012761	061254	000014		MOV	#BUFF3,14(1)		; SET BUFFER #2 ADDR.
5014	024344	012761	060252	000020		MOV	#BUFF2,20(1)		; THE THIRD OUTPUT IS NECESSARY
5015									; TO COMPENSATE FOR SAMPLES LOST
5016	024352	105061	000012			CLRB	12(1)		; NO EXT.
5017	024356	012761	000200	000054		MOV	#200,54(1)		; DELAY 1 TICK.

M11

MAINDEC -11- DRLPA-A
DRLPA.P11 T42

MACY11 27(654) 13-DEC-77 12:58 PAGE 112
*TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DRI1K #3

SEQ 0142

```

5018 024364 012761 000002 000056      MOV      #3-1,56(1)      ;SET CHAN #= DR#-1
5019 024372 012761 000001 000060      MOV      #1,60(1)      ;SAMPLE ONLY ONE CHAN.
5020 024400 012761 000001 000062      MOV      #1,62(1)      ;ONE TICK BETWEEN SAMPLE
5021 024406 005061 000064      CLR      64(1)          ;NO START/EVENT
5022 024412 005061 000066      CLR      66(1)          ;NO START MASK.
5023 024416 005061 000070      CLR      70(1)          ;NO EVENT MASK.
5024
5025 024422 012700 125252      MOV      #125252,R0     ;FILL OUTPUT BUFFER W/PATTERN
5026 024426 012701 001002      MOV      #514,R1
5027 024432 012702 060252      MOV      #BUFF2,R2
5028 024436 010022      4$:      MOV      R0,(2)+
5029 024440 005100      COM      R0
5030 024442 005301      DEC      R1
5031 024444 001374      BNE      4$
5032
5033 024446 012700 056246      MOV      #BUFF0,R0     ;CLEAR INPUT BUFFER
5034 024452 012701 001002      MOV      #514,R1
5035 024456 005020      5$:      CLR      (0)+
5036 024460 005301      DEC      R1
5037 024462 001375      BNE      5$
5038
5039 024464 012737 000011 055066      MOV      #11,JOBO      ;OK,NOW WE GOTTO STOP THE CLOCK
5040 024472 012737 000000 055070      MOV      #0,JOBO+2     ;SO THAT IT DOESN'T START UNTIL
5041 024500 012737 000011 055440      MOV      #11,JOB3      ;DRIK PARAMETER SET UP, THEN AS
5042 024506 012737 000503 055442      MOV      #503,JOB3+2   ;JOB #4, WE START CLOCK.
5043 024514 012737 177000 055444      MOV      #177000,JOB3+4
5044 024522 012737 000004 001124      MOV      #4,$GDDAT     ;FOUR JOBS.
5045 024530 012737 000002 002014      MOV      #2,FUDGE      ;/SET UP A FUDGE FACTOR TO
5046                                     ;/COMPENSATE FOR THE TWO KW11K JOBS
5047 024536 004737 036026      JSR      PC, FLASH     ;/GO DO THEM!
5048
5049 024542 032737 000200 001564      BIT      #BIT7,SR2     ;/CABLED TOGETHER?
5050 024550 001417      BEQ      TST43
5051 024552 012700 001003      MOV      #515,R0
5052 024556 012701 060252      MOV      #BUFF2,R1     ;/NOW CHECK DATA IF CABLED.
5053 024562 012702 056246      MOV      #BUFF0,R2
5054 024566 005300      6$:      DEC      R0
5055 024570 001407      BEQ      TST43
5056 024572 022221      CMP      (R2)+,(R1)+   ;;
5057 024574 001774      BEQ      6$
5058 024576 014137 001124      MOV      -(R1),$GDDAT
5059 024602 014237 001126      MOV      -(R2),$3DDAT
5060
5061 024606 104012      ERROR   12
5062 024610      20$.
5063
5064
5065
5066
5067
5068
5069
5070
5071

```

```

*****
*TEST 43      *TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DRI1K #4
*
*IN THIS TEST WE'LL MAKE SURE THAT DRI1K #4 WILL PASS DATA.
*SR1 BIT8 SELECTS THIS DRI1K FOR TEST. SR2 BIT8 INDICTES
*THAT THE INPUT IS CALLED BACK TO THE OUTPUT.
*

```

```

;* NOTE: THIS TEST IS A MULTI-USER MICRO-CODE TEST ONLY!
;* IF DEDICATE USER MICRO-CODE, THIS TEST WILL BE BY-PASSED.
;* THIS TEST WAS DESIGNED IN ORDER TO FIND A WILD LPA-11 SYSTEM FAULT.
;* I DON'T REALLY EXPECT AN ERROR HERE UNLESS YOU'VE GOT A HEAVY FAULT.
;* ARBITRATION LOGIC IS ONE BIG REASON WHY YOU COULD FAIL. ANOTHER
;* IS THAT INTERRUPT VECTOR COULD BE WRONG. ALSO THE DMC MICRO-CODE MIGHT
;* NOT BE EXECUTING PROPERLY.

```

```

*****

```

```

†ST43: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION

;--DRLT-
;RUNNING MULTI-USER MICRO-CODE?
;YES-CONTINUE.
;NO-EXIT THIS TEST.

1$: MOV #JOB0,RO ;CLEAR OUT THE RDA TABLES
25$: CLR (RO)+
CMP RO,#JOB4
BNE 25$

BIT #BIT8,SRI ;IS THIS DRI1K SELECTED FOR TEST?
BNE 2$ ;YES-TEST IT.
JMP 20$ ;NO-EXIT THIS TEST.

2$: MOV #JOB2,RO ;PICK UP THIS JOB ADDRESS.
MOV #BIT1!BIT3,(0) ;SET START AND MULTI-USER.
BIS #BIT4!BIT8,(0) ;SELECT DIGITAL INPUT SINGLE CHAN.
3$: MOV #257,2(0) ;SET WORD COUNT
MOV #JOB20,4(0) ;USW ADDR.
CLRB 6(0) ;NO EXT.
MOVB #201,7(0) ;MAKE TWO BUFFERS AVAIL.
MOV #BUFF0,10(0) ;1ST BUFFER ADDR.
MOV #BUFF1,14(0) ;2ND BUFFER ADDR.
CLR 12(0) ;NO EXT.
MOV #200,54(0) ;DELAY =200 TICKS.
MOV #4-1,56(0) ;SET CHAN #=DRI1K#-1.
MOV #1,60(0) ;SAMPLING ONLY ONE CHANNEL.
MOV #1,62(0) ;ON PULSE BETWEEN SAMPLE IS NO EXTERNAL TRIG.
CLR 64(0) ;NO START/EVENT MARK WORD
CLR 66(0) ;NO START MASK
CLR 70(0) ;NO EVENT MASK.

MOV #JOB1,R1 ;OK-NOW LETS SET UP THE OUTPUT.
MOV #BIT1!BIT3,(1) ;SET START AND MULTI USER.
BIS #BIT4!BIT7!BIT8,(1) ;SELECT DIGITAL OUT.
MOV #257,2(1) ;WORD COUNT
MOV #JOB10,4(1) ;SET USW ADDR.
CLRB 6(1) ;NO EXT.
MOVB #202,7(1)
MOV #BLFF2,10(1) ;SET BUFFER #1 ADDR.

```

```

5072
5073
5074
5075
5076
5077
5078
5079
5080
5081
5082 024610 000004
5083 024612 012737 000001 001160
5084
5085
5086 024620 122737 000115 001772
5087 024626 001402
5088 024630 000137 025334
5089
5090 024634 012700 055066 1$:
5091 024640 005020 25$:
5092 024642 020027 055556
5093 024646 001374
5094
5095
5096 024650 032737 000400 001562
5097 024656 001002
5098 024660 000137 025334
5099
5100 024664 012700 055322 2$:
5101 024670 012710 000012
5102 024674 052710 000420
5103 024700 012760 000401 000002
5104 024706 012760 055420 000004 3$:
5105 024714 105060 000006
5106 024720 112760 000201 000007
5107 024726 012760 056246 000010
5108 024734 012760 057250 000014
5109 024742 005060 000012
5110 024746 012760 000200 000054
5111 024754 012760 000003 000056
5112 024762 012760 000001 000060
5113 024770 012760 000001 000062
5114 024776 005060 000064
5115 025002 005060 000066
5116 025006 005060 000070
5117
5118 025012 012701 055204
5119 025016 012711 000012
5120 025022 052711 000620
5121 025026 012761 000401 000002
5122 025034 012761 055302 000004
5123 025042 105061 000006
5124 025046 112761 000202 000007
5125 025054 012761 060252 000010

```



```

5126 025062 012761 061254 000014      MOV      #BUFF3,14(1)      ;SET BUFFER #2 ADDR.
5127 025070 012761 060252 000020      MOV      #BUFF2,20(1)      ;THE THIRD OUTPUT IS NECESSARY
5128                                     ;TO COMPENSATE FOR SAMPLES LOST
5129 025076 105061 000012      CLR#     12(1)             ;NO EXT
5130 025102 012761 000200 000054      MOV      #200,54(1)        ;DELAY 1 TICK.
5131 025110 012761 000003 000056      MOV      #4-1,56(1)        ;SET CHAN #= DR#-1
5132 025116 012761 000001 000060      MOV      #1,60(1)          ;SAMPLE ONLY ONE CHAN.
5133 025124 012761 000001 000062      MOV      #1,62(1)          ;ONE TICK BETWEEN SAMPLE
5134 025132 005061 000064      CLR      64(1)             ;NO START/EVENT
5135 025136 005061 000066      CLR      66(1)             ;NO START MASK.
5136 025142 005061 000070      CLR      70(1)             ;NO EVENT MASK.
5137
5138 025146 012700 125252      MOV      #125252,R0        ;FILL OUTPUT BUFFER W/PATTERN
5139 025152 012701 001002      MOV      #514,R1
5140 025156 012702 060252      MOV      #BUFF2,R2
5141 025162 010022 4$:      MOV      R0,(2)+
5142 025164 005100      COM      R0
5143 025166 005301      DEC      R1
5144 025170 001374      BNE      4$
5145
5146 025172 012700 056246      MOV      #BUFF0,R0        ;CLEAR INPUT BUFFER
5147 025176 012701 001002      MOV      #514,R1
5148 025202 005020 5$:      CLR      (0)+
5149 025204 005301      DEC      R1
5150 025206 001375      BNE      5$
5151
5152 025210 012737 000011 055066      MOV      #11,JOBO         ;OK,NOW WE GOTTO STOP THE CLOCK
5153 025216 012737 000000 055070      MOV      #0,JOBO+2        ;SO THAT IT DOESN'T START UNTIL
5154 025224 012737 000011 055440      MOV      #11,JOB3         ;DRI1K PARAMETER SET UP, THEN AS
5155 025232 012737 000503 055442      MOV      #503,JOB3+2      ;JOB #4, WE START CLOCK.
5156 025240 012737 177000 055444      MOV      #177000,JOB3+4
5157 025246 012737 000004 001124      MOV      #4,$GDDAT        ;FOUR JOBS.
5158 025254 012737 000002 002014      MOV      #2,$FUDGE        ;/SET UP A FUDGE FACTOR TO
5159                                     ;/COMPENSATE FOR THE TWO KW11K JOBS
5160 025262 004737 036026      JSR PC, FLASH            ;/GO DO THEM!
5161
5162 025266 032737 000400 001564      BIT      #BIT8,SR2        ;/CABLED TOGETHER?
5163 025274 001417      BEQ      TST44            ;
5164 025276 012700 001003      MOV      #515,R0         ;/NOW CHECK DATA IF CABLED.
5165 025302 012701 060252      MOV      #BUFF2,R1
5166 025306 012702 056246      MOV      #BUFF0,R2
5167 025312 005000 6$:      DEC      R0
5168 025314 001407      BEQ      TST44            ;
5169 025316 022221      CMP      (R2)+,(R1)+      ;
5170 025320 001774      BEQ      6$
5171 025322 014137 001124      MOV      -(R1),$GDDAT
5172 025326 014237 001126      MOV      -(R2),$BDDAT
5173
5174 025332 104012 20$:      ERROR 12
5175 025334
5176
5177
5178
5179
;*****
;*TEST 44 *TEST THAT THE MULTI-USER MICRO-CODE WILL EXERCISE DRI1K #5

```

```

5180
5181
5182
5183
5184
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195 025334 000004
5196 025336 012737 000001 001160
5197
5198
5199 025344 122737 000115 001772
5200 025352 001402
5201 025354 000137 026060
5202
5203 025360 012700 055066
5204 025364 005020
5205 025366 020027 055556
5206 025372 001374
5207
5208
5209 025374 032737 001000 001562
5210 025402 001002
5211 025404 000137 026060
5212
5213 025410 012700 055322
5214 025414 012710 000012
5215 025420 052710 000420
5216 025424 012760 000401 000002
5217 025432 012760 055420 000004
5218 025440 105060 000006
5219 025444 112760 000201 000007
5220 025452 012760 056246 000010
5221 025460 012760 057250 000014
5222 025466 005060 000012
5223 025472 012760 000200 000054
5224 025500 012760 000004 000056
5225 025506 012760 000001 000060
5226 025514 012760 000001 000062
5227 025522 005060 000064
5228 025526 005060 000066
5229 025532 005060 000070
5230
5231 025536 012701 055204
5232 025542 012711 000012
5233 025546 052711 000620

```

```

*
*IN THIS TEST WE'LL MAKE SURE THAT DRI1K #5 WILL PASS DATA.
*SRI BIT9 SELECTS THIS DRI1K FOR TEST. SR2 BIT9 INDICATES
*THAT THE INPUT IS CALLED BACK TO THE OUTPUT.
*
* NOTE: THIS TEST IS A MULTI-USER MICRO-CODE TEST ONLY!
* IF DEDICATE USER MICRO-CODE, THIS TEST WILL BE BY-PASSED.
*
*THIS TEST WAS DESIGNED IN ORDER TO FIND A WILD LPA-11 SYSTEM FAULT.
*I DON'T REALLY EXPECT AN ERROR HERE UNLESS YOU'VE GOT A HEAVY FAULT.
*ARBITRATION LOGIC IS ONE BIG REASON WHY YOU COULD FAIL. ANOTHER
*IS THAT INTERRUPT VECTOR COULD BE WRONG. ALSO THE DMC MICRO-CODE MIGHT
*NOT BE EXECUTING PROPERLY.
*
*****
↑ST44: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
; /-DRLT-
; RUNNING MULTI-USER MICRO-CODE?
; YES-CONTINUE.
; NO-EXIT THIS TEST.
1$: MOV #JOB0,RO ;CLEAR OUT THE RDA TABLES
25$: CLR (RO)+
CMP RO,#JOB4
BNE 25$
BIT #BIT9,SR1 ;IS THIS DRI1K SELECTED FOR TEST?
BNE 2$ ;YES-TEST IT.
JMP 20$ ;NO-EXIT THIS TEST.
2$: MOV #JOB2,RO ;PICK UP THIS JOB ADDRESS.
MOV #BIT1!BIT3,(0) ;SET START AND MULTI-USER.
BIS #BIT4!BIT8,(0) ;SELECT DIGITAL INPUT SINGLE CHAN.
3$: MOV #257,2(0) ;SET WORD COUNT
MOV #JOB20,4(0) ;USW ADDR.
CLRB 6(0) ;NO EXT.
MOVB #201,7(0) ;MAKE TWO BUFFERS AVAIL.
MOV #BUFF0,10(0) ;1ST BUFFER ADDR.
MOV #BUFF1,14(0) ;2ND BUFFER ADDR.
CLR 12(0) ;NO EXT.
MOV #200,54(0) ;DELAY =200 TICKS.
MOV #5-1,56(0) ;SET CHAN #=DRI1K#-1.
MOV #1,60(0) ;SAMPLING ONLY ONE CHANNEL.
MOV #1,62(0) ;ON PULSE BETWEEN SAMPLE IS NO EXTERNAL TRIG.
CLR 64(0) ;NO START/EVENT MAKR WORD
CLR 66(0) ;NO START MASK
CLR 70(0) ;NO EVENT MASK.
MOV #JOB1,R1 ;OK-NOW LETS SET UP THE OUTPUT.
MOV #BIT1!BIT3,(1) ;SET START AND MULTI USER.
BIS #BIT4!BIT7!BIT8,(1) ;SELECT DIGITAL OUT.

```

5234	025552	012761	000401	000002		MOV	#257.,2(1)	;WORD COUNT
5235	025560	012761	055302	000004		MOV	#JOB10,4(1)	;SET USW ADDR.
5236	025566	105061	000006			CLRB	6(1)	;NO EXT.
5237	025572	112761	000202	000007		MOV	#202,7(1)	
5238	025600	012761	060252	000010		MOV	#BUFF2,10(1)	;SET BUFFER #1 ADDR.
5239	025606	012761	061254	000011		MOV	#BUFF3,14(1)	;SET BUFFER #2 ADDR.
5240	025614	012761	060252	000020		MOV	#BUFF2,20,1)	;THE THIRD OUTPUT IS NECESSARY
5241								;TO COMPENSATE FOR SAMPLES LOST
5242	025622	105061	000012			CLRB	12(1)	;NO EXT.
5243	025626	012761	000200	000054		MOV	#200,54(1)	;DELAY 1 TICK.
5244	025634	012761	000004	000056		MOV	#5-1,56(1)	;SET CHAN #= DR#-1
5245	025642	012761	000001	000060		MOV	#1,60(1)	;SAMPLE ONLY ONE CHAN.
5246	025650	012761	000001	000062		MOV	#1,62(1)	;ONE TICK BETWEEN SAMPLE
5247	025656	005061	000064			CLR	64(1)	;NO START/EVENT
5248	025662	005061	000066			CLR	66(1)	;NO START MASK.
5249	025666	005061	000070			CLR	70(1)	;NO EVENT MASK.
5250								
5251	025672	012700	125252			MOV	#125252,R0	;FILL OUTPUT BUFFER W/PATTERN
5252	025676	012701	001002			MOV	#514.,R1	
5253	025702	012702	060252			MOV	#BUFF2,R2	
5254	025706	010022			4\$:	MOV	R0,(2)+	
5255	025710	005100				COM	R0	
5256	025712	005301				DEC	R1	
5257	025714	001374				BNE	4\$	
5258								
5259	025716	012700	056246			MOV	#BUFF0,R0	;CLEAR INPUT BUFFER
5260	025722	012701	001002			MOV	#514.,R1	
5261	025726	005020			5\$:	CLR	(0)+	
5262	025730	005301				DEC	R1	
5263	025732	001375				BNE	5\$	
5264								
5265	025734	012737	000011	055066		MOV	#11,JOB0	;OK,NOW WE GOTTO STOP THE CLOCK
5266	025742	012737	000000	055070		MOV	#0,JOB0+2	;SO THAT IT DOESN'T START UNTIL
5267	025750	012737	000011	055440		MOV	#11,JOB3	;DRIIK PARAMETER SET UP, THEN AS
5268	025756	012737	000503	055442		MOV	#503,JOB3+2	;JOB #4, WE START CLOCK.
5269	025764	012737	177000	055444		MOV	#177000,JOB3+4	
5270	025772	012737	000004	001124		MOV	#4,\$GDDAT	;FOUR JOBS.
5271	026000	012737	000002	002014		MOV	#2,\$FUDGE	;SET UP A FUDGE FACTOR TO
5272								;COMPENSATE FOR THE TWO KWIIK JOBS
5273	026006	004737	036026			JSR PC, FLASH		;GO DO THEM!
5274								
5275	026012	032737	001000	001564		BIT	#BIT9,SR2	;CABLED TOGETHER?
5276	026020	001417				BEQ	TST45	
5277	026022	012700	001003			MOV	#515.,R0	;NOW CHECK DATA IF CABLED.
5278	026026	012701	060252			MOV	#BUFF2,R1	
5279	026032	012702	056246			MOV	#BUFF0,R2	
5280	026036	005300			6\$:	DEC	R0	
5281	026040	001407				BEQ	TST45	;;
5282	026042	022221				CMP	(R2)+,(R1)+	
5283	026044	001774				BEQ	6\$	
5284	026046	014137	001124			MOV	-(R1),\$GDDAT	
5285	026052	014237	001126			MOV	-(R2),\$BDDAT	
5286								
5287	026056	104012				ERROR	12	

5288 026060

20\$:

```

*****
*TEST 45 *TEST THE LPA SYSTEM USER MICRO-CODE'S ABILITY TO TAKE SAMPLE SINGLE JOB
*
*IN THIS TEST WE'LL VERIFY ANOLOG SAMPLING ABILITIES.
*DATA WILL BE TAKEN ON ALL CHANNELLS THROUGH ONE JOB.
*AT THE CONCLUSION, IF A WRAP AROUND MODULE WAS ON THE ANOLOG
*MODULE, A CHECK WILL BE MADE ON SPECIFIC CHANNELLS
*FOR SPECIFIC VALUES.
*
*THIS TEST WAS DESIGNED IN ORDER TO FIND A WILD LPA-11 SYSTEM FAULT.
*I DON'T REALLY EXPECT AN ERROR HERE UNLESS YOU'VE GOT A HEAVY FAULT.
*ARBITRATION LOGIC IS ONE BIG REASON WHY YOU COULD FAIL. ANOTHER
*IS THAT INTERRUPT VECTOR COULD BE WRONG. ALSO THE DMC MICRO-CODE MIGHT
*NOT BE EXECUTING PROPERLY.
*
*****

```

5309 026060 000004

†ST45: SCOPE

```

5310
5311 026062 032737 012001 001562
5312 026070 001002
5313 026072 000137 026416
5314 026076 012737 170000 044116 1$:
5315 026104 012777 044112 153364
5316 026112 052777 000001 153346
5317 026120 012700 055066
5318 026124 005020 10$:
5319 026126 022700 056230
5320 026132 001374
5321
5322 026134 012700 055066
5323 026140 012710 000002
5324 026144 052710 001000 2$:
5325 026150 122737 000104 001772
5326 026156 001402
5327 026160 052710 000010
5328 026164 012760 000401 000002 3$:
5329 026172 012760 055664 000004
5330 026200 005070 000004
5331 026204 012760 056246 000010
5332 026212 012760 057250 000014
5333 026220 012760 060252 000020
5334 026226 112760 000002 000007
5335 026234 122737 000104 001772
5336 026242 001003
5337 026244 152760 000200 000007
5338
5339 026252 31$:
5340 026252 012760 000144 000054
5341 026260 012760 000020 000060

```

```

BIT #BIT12!BIT10!BIT0,SRI ;IS THERE AN ANOLOG DEVICE?
BNE 1$
JMP 20$
MOV #170000,KWT+4 ;RESTART CLOCK AT 1 MEGHZ RATE
MOV #KWT,ALPADL
BIS #BIT0,ALPCI
MOV #JOB0,RO ;CLEAR OUT JOB AREA.
CLR (0)+
CMP #JOB7R,RO
BNE 10$
MOV #JOB0,RO ;GET SET TO JO SINGLE JOB RDA
MOV #2,(0)
BIS #BIT9,(0) ;SET SEQUENTIAL CHAN SAMPLE
CMPB #'D,$VERSN ;MULTI USER MICRO-CODE?
BEQ 3$
BIS #BIT3,(0) ;YES-TELL RDA MODE WORD.
MOV #257,2(0) ;SET BUFFER SIZE.
MOV #JOB0U,4(0) ;SET USW WORD.
CLR 24(0) ;CLEAR USW WORD.
MOV #BUFF0,10(0) ;NOW SET BUFFER ADDRESS
MOV #BUFF1,14(0)
MOV #BUFF2,20(0)
MOV #2,7(0) ;INDICATE HOW MANY BUFFERS USED.
CMPB #'D,$VERSN ;IF THIS IS DEDICATED MODE,
BNE 31$ ;WE MUST LET BUFFER OVERRUN BE NON-FATAL.
BISB #BIT7,7(0) ;DEDICATED MODE IS JUST TO FAST FOR
;THE SIZE BUFFER AND STYLE WE USE HERE.
MOV #100.,54(0) ;SET DELAY BEFORE START?
MOV #16.,60(0) ;SET # OF CHANNELS

```

```

5342 026266 012760 000001 000062 4$: MOV #+1,62(0) ;# OF TICKS BETWEEN SAMPLES
5343 026274 012760 000400 000056 MOV #400,56(0) ;START CH 0, INC=1
5344 026302 016037 000060 034454 MOV 60(0),CHANU
5345 026310 012737 000001 001124 MOV #1,$GDOAT ;ONLY ONE JOB.
5346 026316 004737 036026 JSR PC,FLASH ;GO-DO IT.
5347 026322 032737 002021 001564 BIT #BIT0!BIT4!BIT10,SR2 ;G5034 WRAPAROUND MODULE?
5348 026330 001432 BEQ TST46 ;
5349 ;YES-LETS CHECK OUT THE RESULTS!
5350 026332 005237 034540 INC REPOR
5351 026336 012700 055066 MOV #JOB0,R0
5352 026342 012703 055006 MOV #LIST10,R3 ;GET LIST OF S/B RESULTS FOR 10 BITS.
5353 026346 032737 002000 001562 BIT #BIT10,SR1 ;SEC IF 12 BIT A/D
5354 026354 001002 BNE 5$
5355 026356 012703 055036 MOV #LIST12,R3 ;GET LIST OF 12 BIT A/D RESULTS.
5356 026362 012702 000004 5$: MOV #4.,R2 ;CHECK 4 CHANS
5357 026366 011337 034474 6$: MOV (3),CHANS ;PICK UP CHAN #
5358 026372 012337 034476 MOV (3)+,CHANF ;
5359 026376 012337 034514 MOV (3)+,AVEXP ;
5360 026402 012337 034512 MOV (3)+,TOLER ;
5361 026406 004737 034146 JSR PC,AVERR
5362 026412 005302 DEC R2
5363 026414 001364 BNE 6$

```

```

5364 026416 20$:
5365 ;*****
5366 ;*TEST 46 *TEST THE LPA SYSTEM USER MICRO-CODE'S ABILITY TO TAKE RANDOM SAMPLES SI
5367 ;*
5368 ;*IN THIS TEST WE'LL VERIFY ANOLOG SAMPLING ABILITIES.
5369 ;*DATA WILL BE TAKEN AN ALL CHANNELLS THROUGH ONE JOB.
5370 ;*
5371 ;*THE DIFFERENCE BETWEEN THIS TEST AND THE ONE BEFORE IF IS THAT
5372 ;*DATA IS TAKEN IN RANDOM CHANNELL MODE INSTEAD OF SEQUENCAL.
5373 ;*
5374 ;*AT THE CONCLUSION, IF A WRAP AROUND MODULE WAS ON THE ANOLOG
5375 ;*MODULE, A CHECK WILL BE MADE ON SPECIFIC CHANNELLS
5376 ;*FOR SPECIFIC VALUES.
5377 ;*
5378 ;*THIS TEST WAS DESIGNED IN ORDER TO FIND A WILD LPA-11 SYSTEM FAULT.
5379 ;*I DON'T REALLY EXPECT AN ERROR HERE UNLESS YOU'VE GOT A HEAVY FAULT.
5380 ;*ARBITRATION LOGIC IS ONE BIG REASON WHY YOU COULD FAIL. ANOTHER
5381 ;*IS THAT INTERRRUPT VECTOR COULD BE WRONG. ALSO THE DMC MICRO-CODE MIGHT
5382 ;*NOT BE EXECUTING PROPERLY.
5383 ;*
5384 ;*****
5385 ;*
5386 ;*
5387 ;*
5388 026416 000004 †ST46: SCOPE
5389 ;*
5390 026420 032737 012001 001562 BIT #BIT12!BIT10!BIT0,SR1 ;IS THERE AN ANOLOG DEVICE?
5391 026426 001002 BNE 1$
5392 026430 000137 027014 JMP 20$
5393 026434 012737 170000 044116 1$: MOV #170000,KWT+4 ;RESTART CLOCK AT 1 MEGHZ RATE
5394 026442 012777 044112 153026 MOV #KWT,ALPADL
5395 026450 052777 000001 153010 BIS #BIT0,ALPCI

```

5396	026456	012700	055066		MOV	#JOB0,RO	;CLEAR OUT JOB AREA.
5397	026462	005020		10\$:	CLR	(0)+	
5398	026464	022700	056230		CMP	#JOB7R,PO	
5399	026470	001374			BNE	10\$	
5400							
5401	026472	012700	055066		MOV	#JOB0,RO	;GET SET TO #0 SINGLE JOB RDA
5402	026476	012710	000002		MOV	#2,(0)	
5403	026502			2\$:			
5404	026502	122737	000104	001772	CMPB	#'D,\$VERSN	;MULTI USER MICRO-CODE?
5405	026510	001402			BEQ	3\$	
5406	026512	052710	000010		BIS	#BIT3,(0)	;YES-TELL RDA MODE WORD.
5407	026516	012760	000401	000002	MOV	#257,2(0)	;SET BUFFER SIZE.
5408	026524	012760	055164	000004	MOV	#JOB0U,4(0)	;SET USW WORD.
5409	026532	005070	000004		CLR	24(0)	;CLEAR USW WORD.
5410	026536	012760	056246	000010	MOV	#BUFF0,10(0)	;NOW SET BUFFER ADDRESS
5411	026544	012760	057250	000014	MOV	#BUFF1,14(0)	
5412	026552	012760	060252	000020	MOV	#BUFF2,20(0)	
5413	026560	112760	000002	000007	MOVB	#2,7(0)	;INDICATE HOW MANY BUFFERS USED.
5414	026566	122737	000104	001772	CMPB	#'D,\$VERSN	;IF THIS IS DEDICATED MODE,
5415	026574	001003			BNE	31\$;WE MUST LET BUFFER OVERRUN BE NON-FATEL.
5416	026576	152760	000200	000007	BISB	#BIT7,7(0)	;DEDICATED MODE IS JUST TO FAST FOR
5417							;THE SIZE BUFFER AND STYLE WE USE HERE.
5418	026604						
5419	026604	012760	000144	000054	MOV	#100,54(0)	;SET DELAY BEFORE START
5420	026612	012760	000007	000060	MOV	#7,60(0)	;SET # OF CHANNELS
5421	026620	012760	000010	000062	MOV	#+10,62(0)	;# OF TICKS BETWEEN SAMPLES
5422	026626	012760	055166	000050	MOV	#JOB0R,50(0)	;SET ADDR OF RANDOM CHANNEL LIST
5423	026634	016037	000060	034454	MOV	50(0),CHANU	
5424	026642	012700	055166		MOV	#JOB0R,RO	;PICK UP POINT TO RANDOM LIST.
5425	026646	012720	000000		MOV	#0,(0)+	;CHAN 0 FIRST
5426	026652	012720	000003		MOV	#3,(0)+	;NEXT CHAN 3
5427	026656	012720	000002		MOV	#2,(0)+	;CHAN 2 NEXT
5428	026662	012720	000003		MOV	#3,(0)+	;CHAN 3
5429	026666	012720	000004		MOV	#4,(0)+	;NEXT CHAN 4
5430	026672	012720	000015		MOV	#15,(0)+	;CHAN 15
5431	026676	012710	000003		MOV	#3,(0)	;CHAN 3 NEXT.
5432	026702	052720	140000		BIS	#BIT15:BIT14,(0)	;NOP AND END OF LIST.
5433	026706	012737	000001	001124	MOV	#1,\$GDDAT	;ONLY ONE JOB.
5434	026714	004737	036026		JSR	PC,FLASH	;GO-DO IT.
5435	026720	032737	002021	001564	BIT	#BIT0:BIT4:BIT10	;SR2 ;G5034 WRAPAROUND MODULE?
5436	026726	001432			BEQ	TST47	
5437							;YES-LETS CHECK OUT THE RESULTS!
5438	026730	005237	034540		INC	REPOR	
5439	026734	012700	055066		MOV	#JOB0,RO	
5440	026740	012703	055006		MOV	#LIST10,R3	;GET LIST OF S/B RESULTS FOR 10 BITS.
5441	026744	032737	002000	001562	BIT	#BIT10,SRI	;SEC IF 12 BIT A/D
5442	026752	001002			BNE	5\$	
5443	026754	012703	055036		MOV	#LIST12,R3	;GET LIST OF 12 BIT A/D RESULTS.
5444	026760	012702	000004		MOV	#4,R2	;CHECK 4 CHANS
5445	026764	011337	034474		MOV	(3),CHANS	;PICK UP CHAN #
5446	026770	012337	034476		MOV	(3)+,CHANF	
5447	026774	012337	034514		MOV	(3)+,AVEXP	
5448	027000	012337	034512		MOV	(3)+,TOLER	
5449	027004	004737	034146		JSR	PC,AVERR	

5450 027010 005302
5451 027012 001364
5452
5453 027014
5454

DEC R2
BNE 65

20\$:

*TEST 47 *TEST THAT MULTI JOBS CAN BE STARTED AND RUN
*IN THIS TEST, WE'LL ATTEMPT TO STARTUP UP TO EIGHT JOBS WITH
*MULTIUSER MICRO CODE. THIS TEST WILL ONLY BE EXECUTED IF
*MULTIUSER MICRO-CODE HAS BEEN LOADED INTO THE KMC.
*LESS THAN EIGHT JOBS MAY BE RUN, DEPENDING UPON THE
*PARTICULAR CONFIGURATION OF YOUR SYSTEM.
*BELOW IS A LIST OF JOBS AND HOW THEY ARE SET UP.

*
* JOB#0 AD11K#1 (IF CONFIGURED) OR LPS A/D OR AR11
* JOB#1 DR11K#1 (IF CONFIGURED) OR LPS I/O OR AR11 I/O
* JOB#2 AR11K (IF CONFIGURED) OR LPS OR AR11
* JOB#3 AD11K#2 (IF CONFIGURED)
* JOB#4 DR11K#2 (IF CONFIGURED)
* JOB#5 DR11K#3 (IF CONFIGURED)
* JOB#6 DR11K#4 (IF CONFIGURED)
* JOB#7 DR11K#5 (IF CONFIGURED)
*

*WHEN WE SET UP THESE JOBS, WE'LL SET THEM UP TO START ROUGHLY
*AT THE SAME TIME BY VARIING THE START UP DELAY.

*WHAT WE'RE GONNA LOOK FOR AN ERROR INDICATION
*IS FOR ERRORS ON JOBS, OR SOME JOB TO FAIL TO FINISH.
*

5481
5482 027014 000004
5483 027016 012737 000001 001160
5484 027024 122737 000115 001772
5485 027032 001402
5486 027034 000137 030220
5487 027040 012700 055066
5488 027044 012703 055204
5489 027050 160003
5490 027052 005020
5491 027054 020027 056230
5492 027060 001374
5493 027062 005037 001124
5494 027066 032737 012001 001562
5495 027074 001442
5496 027076 012700 055066
5497 027102 012710 000012
5498 027106 052710 000400
5499 027112 012760 000512 000002
5500 027120 012760 055164 000004
5501 027126 005070 000004
5502 027132 152760 000200 000007
5503 027140 012760 056246 000010

*ST47: SCOPE
MOV #1,\$TIMES ;;DO 1 ITERATION
CMPB #'M,\$VERSN
BEQ 1\$
JMP 20\$
1\$: MOV #JOB0,R0
MOV #JOB1,R3 ;GET DIF IN ADDR. BETWEEN JOBS.
SUB R0,R3
2\$: CLR (0)+
CMP R0,#JOB7R
BNE 2\$
CLR \$GDDAT
BIT #BIT0!BIT10!BIT12,SRI ;ANY A/D'S?
BEQ 3\$;NO-EXIT A/D'S.
MOV #JOB0,R0 ;GET JOB POINTER.
MOV #12,(0) ;MODE WORD-START A/D.
BIS #BIT8,(0) ;SINGLE CHAN.
MOV #512,2(0) ;TAKE 512 SAMPLES
MOV #JOB0U,4(0)
CLR 24(0) ;CLEAR USW
BISB #BIT7,7(0) ;BUFFER OVERRUN NOT FATAL. ONE BUFFER.
MOV #BUFF0,10(0) ;SET BUFFER ADDR.

```

5504 027146 012760 000010 000054      MOV      #10,54(0)      ;SET DELAY BEFORE START.
5505 027154 012760 000016 000060      MOV      #16,60(0)      ;SAMPLE 14 CHANNELS
5506 027162 012760 000010 000062      MOV      #10,62(0)      ;SET SAMPLE RATE.
5507 027170 005237 001124      INC      $GDDAT
5508 027174 012760 000400 000056      MOV      #400,56(0)    ;START CH0, INC=1
5509 027202
5510
5511
5512
5513
5514 027202 032737 100004 001562      BIT      #BIT2!BIT15,SRI ;/THIS DR11K #1 OR LPS I/O SEL?
5515 027210 001442      BEQ      64$
5516 027212 060300      ADD      R3,R0          ;/UPDAT JOB STORAGE AREA POINTER.
5517 027214 012710 000432      MOV      #432,(0)      ;/SET I/O START, MULTI-USER, INPUT SINGLE CHAN.
5518 027220 012760 000256 000002      MOV      #256,2(0)    ;/WORD COUNT
5519 027226 012760 055302 000004      MOV      #JOB1U,4(0)
5520 027234 012770 000000 000004      MOV      #0,24(0)     ;/CLEAR USW
5521 027242 012760 000000 000006      MOV      #0,6(0)
5522 027250 012760 060252 000010      MOV      #BUFF2,10(0) ;/SET BUFFER ADDR.
5523 027256 005060 000012      CLR      12(0)
5524 027262 005237 001124      INC      $GDDAT        ;RECORD THIS JOB
5525 027266 012760 000100 000054      MOV      #100,54(0)   ;DELAY BEFORE START.
5526 027274 012760 000000 000056      MOV      #1-1,56(0)  ;/SELECT CHAN #.
5527 027302 012760 000001 000060      MOV      #1,60(0)    ;/SAMPLING ONLY ONE CHAN.
5528 027310 012760 000001 000062      MOV      #1,62(0)    ;/SAMPLE RATE
5529
5530 027316
5531
5532 027316 032737 042010 001562      BIT      #BIT3!BIT10!BIT14,SRI ;D/A OUTPUT SELECTED?
5533 027324 001440      BEQ      5$           ;NO-NEXT SET-UP
5534 027326 005237 001124      INC      $GDDAT
5535
5536
5537
5538
5539 027332 060300      ADD      R3,R0          ;ALTHOUGH WE'RE GONNA WORK
5540 027334 012710 001212      MOV      #BIT1!BIT3!BIT7!BIT9,(0) ;WITH D/A, NO OUTPUT
5541 027340 012760 000256 000002      MOV      #256,2(0)   ;WILL RESULT. D/A THROWN IN
5542 027346 012760 055420 000004      MOV      #JOB2U,4(0) ;JUST TO TRY AND CONFUSE LPA.
5543 027354 012770 000000 000004      MOV      #0,24(0)    ;UPDATE JOB STORAGE AREA POINTER
5544 027362 012760 000000 000006      MOV      #0,6(0)     ;START D/A, SEQ. CHAN.
5545 027370 012760 061254 000010      MOV      #BUFF3,10(0) ;SET XFERR COUNT
5546 027376 012760 000100 000054      MOV      #100,54(0)  ;CLR USW.
5547 027404 012760 000000 000056      MOV      #0,56(0)    ;BUFFER ADDR.
5548 027412 012760 000001 000060      MOV      #1,60(0)   ;DELAY BEFORE START.
5549 027420 012760 000001 000062      MOV      #1,62(0)   ;/SAMPLE CHAN 0
5550
5551 027426 032737 000020 001562      BIT      #BIT4,SRI    ;/SAMPLE ONLY ONE CHAN
5552 027434 001437      BEQ      5$          ;/SAMPLE RATE
5553 027436 060300      ADD      R3,R0          ;SECOND A/D?
5554 027440 012710 000012      MOV      #12,(0)     ;UPDATE JOB STORAGE ARE POINTER.
5555 027444 012760 000512 000002      MOV      #512,2(0)  ;MODE WORD-START A/D.
5556 027452 012760 055536 000004      MOV      #JOB3U,4(0) ;TAKE 512 SAMPLES
5557 027460 005070 000004      CLR      24(0)       ;CLEAR USW

```



```

5558 027464 152760 000200 000007      BISB      #BIT7,7(0)      ;BUFFER OVERRUN NOT FATAL, ONE BUFFER.
5559 027472 012760 056246 000010      MOV      #BUFF0,10(0) ;SET BUFFER ADDR.
5560 027500 012760 000010 000054      MOV      #10,54(0)  ;SET DELAY BEFORE START.
5561 027506 012760 000420 000056      MOV      #420,56(0)
5562 027514 012760 000016 000060      MOV      #16,60(0)  ;SAMPLE 14 CHANNELS.
5563 027522 012760 000010 000062      MOV      #10,62(0)  ;SET SAMPLE RATE.
5564 027530 005237 001124      INC      $GDDAT
5565
5566 027534          65:
5567
5568
5569
5570 027534 032737 000040 001562      BIT      #BITS,SR1 ;/-SDRM-
5571 027542 001442      BEQ      65$      ;/THIS DR11K #2 SELECTED?
5572 027544 060300      ADD      R3,R0    ;/UPDAT JOB STORAGE AREA POINTER.
5573 027546 012710 000432      MOV      #432,(0) ;/SET I/O START, MULTI-USER, INPUT SINGLE CHAN.
5574 027552 012760 000256 000002      MOV      #256,2(0) ;/WORD COUNT
5575 027560 012760 055654 000004      MOV      #JOB4U,4(0)
5576 027566 012770 000000 000004      MOV      #0,24(0)  ;/CLEAR USW
5577 027574 012760 000000 000006      MOV      #0,6(0)
5578 027602 012760 060252 000010      MOV      #BUFF2,10(0) ;/SET BUFFER ADDR.
5579 027610 005060 000012      CLR      12(0)
5580 027614 005237 001124      INC      $GDDAT
5581 027620 012760 000100 000054      MOV      #100,54(0) ;RECORD THIS JOB
5582 027626 012760 000001 000056      MOV      #2-1,56(0) ;DELAY BEFORE START.
5583 027634 012760 000001 000060      MOV      #1,60(0)  ;/SELECT CHAN #.
5584 027642 012760 000001 000062      MOV      #1,62(0)  ;/SAMPLING ONLY ONE CHAN.
5585
5586 027650          65$:
5587
5588
5589
5590 027650 032737 000200 001562      BIT      #BIT7,SR1 ;/-SDRM-
5591 027656 001442      BEQ      66$      ;/THIS DR11K #3 SELECTED?
5592 027660 060300      ADD      R3,R0    ;/UPDAT JOB STORAGE AREA POINTER.
5593 027662 012710 000432      MOV      #432,(0) ;/SET I/O START, MULTI-USER, INPUT SINGLE CHAN.
5594 027666 012760 000256 000002      MOV      #256,2(0) ;/WORD COUNT
5595 027674 012760 055772 000004      MOV      #JOB5U,4(0)
5596 027702 012770 000000 000004      MOV      #0,24(0)  ;/CLEAR USW
5597 027710 012760 000000 000006      MOV      #0,6(0)
5598 027716 012760 060252 000010      MOV      #BUFF2,10(0) ;/SET BUFFER ADDR.
5599 027724 005060 000012      CLR      12(0)
5600 027730 005237 001124      INC      $GDDAT
5601 027734 012760 000100 000054      MOV      #100,54(0) ;RECORD THIS JOB
5602 027742 012760 000002 000056      MOV      #3-1,56(0) ;DELAY BEFORE START.
5603 027750 012760 000001 000060      MOV      #1,60(0)  ;/SELECT CHAN #.
5604 027756 012760 000001 000062      MOV      #1,62(0)  ;/SAMPLING ONLY ONE CHAN.
5605
5606 027764          66$:
5607
5608
5609
5610 027764 032737 000400 001562      BIT      #BIT8,SR1 ;/-SDRM-
5611 027772 001442      BEQ      67$      ;/THIS DR11K #4 SELECTED?

```

```

5612 027774 060300          ADD      R3,RO          ;/UPDAT JOB STORAGE AREA POINTER.
5613 027776 012710 000432    MOV      #432,(0)      ;/SET I/O START, MULTI-USER, INPUT SINGLE CHAN.
5614 030002 012760 000256 000002    MOV      #256,2(0)    ;/WORD COUNT
5615 030010 012760 056110 000004    MOV      #JOB#U,4(0)  ;/CLEAR USW
5616 030016 012770 000000 000004    MOV      #0,24(0)
5617 030024 012760 000000 000006    MOV      #0,6(0)
5618 030032 012760 060252 000010    MOV      #BUFF2,10(0) ;/SET BUFFER ADDR.
5619 030040 005060 000012    CLR      12(0)
5620 030044 005237 001124    INC      $GDDAT       ;RECORD THIS JOB
5621 030050 012760 000100 000054    MOV      #100,54(0)   ;DELAY BEFORE START.
5622 030056 012760 000003 000056    MOV      #4-1,56(0)  ;/SELECT CHAN #.
5623 030064 012760 000001 000060    MOV      #1,60(0)    ;/SAMPLING ONLY ONE CHAN.
5624 030072 012760 000001 000062    MOV      #1,62(0)    ;/SAMPLE RATE
5625
5626 030100          b7$:
5627
5628          ;/--SDRM-
5629
5630 030100 032737 001000 001562    BIT      #BIT9,SRI    ;/THIS DR11K #5 SELECTED?
5631 030106 001442    BEQ      68$
5632 030110 060300          ADD      R3,RO          ;/UPDAT JOB STORAGE AREA POINTER.
5633 030112 012710 000432    MOV      #432,(0)      ;/SET I/O START, MULTI-USER, INPUT SINGLE CHAN.
5634 030116 012760 000256 000002    MOV      #256,2(0)    ;/WORD COUNT
5635 030124 012760 056226 000004    MOV      #JOB#U,4(0)  ;/CLEAR USW
5636 030132 012770 000000 000004    MOV      #0,24(0)
5637 030140 012760 000000 000006    MOV      #0,6(0)
5638 030146 012760 060252 000010    MOV      #BUFF2,10(0) ;/SET BUFFER ADDR.
5639 030154 005060 000012    CLR      12(0)
5640 030160 005237 001124    INC      $GDDAT       ;RECORD THIS JOB
5641 030164 012760 000100 000054    MOV      #100,54(0)   ;DELAY BEFORE START.
5642 030172 012760 000004 000056    MOV      #5-1,56(0)  ;/SELECT CHAN #.
5643 030200 012760 000001 000060    MOV      #1,60(0)    ;/SAMPLING ONLY ONE CHAN.
5644 030206 012760 000001 000062    MOV      #1,62(0)    ;/SAMPLE RATE
5645
5646 030214          68$:
5647
5648 030214 004737 036026    JSR      PC,FLASH      ;FLASH WILL START JOBS
5649          ;AND REPORT ERRORS, WAIT TILL DONE.
5650 030220          20$:
5651
5652          ;*****
5653          ;*TEST 50          *HIGH SPEED A/D SAMPLE TEST (DEDICATED MODE,SPECIAL TEST
5654          ;*
5655          ;*          IN THIS TEST WE WILL TAKE A/D SAMPLES AS FAST AS WE CAN.
5656          ;*          THIS TEST REQUIRES MORE MEMORY THAN THE REST SO UPON ENTRY
5657          ;*          WE WILL CHECK TO MAKE SURE THAT WE HAVE 20K OF MEMORY.
5658          ;*          ALSO DEDICATED MODE MICRO-CODE MUST BE IN THE KMC. LAST,
5659          ;*          WE MUST HAVE TWO A/D S ON THE I/O BUSS,BOTH HAVEING A G5036
5660          ;*          WRAP-ARROUND MODULE INSTALLED AND INFORMED TO PROGRAM VIA SR2.
5661          ;*          IF ALL REQUIREMENTS ARE MENT,WE WILL DO THIS TEST AND TAKE
5662          ;*          SAMPLES AT A RATE OF APP. 150 KILOHERTZ.
5663          ;*
5664          ;*****
5665 030220 000004    †ST50: SCOPE

```

```

5666
5667 030222 122737 000104 001772      CMPB   #'D,$VERSN      ;RUNNING DEDICATED MODE U-CODE?
5668 030230 001014                      BNE    ETDF            ;NO-THEN EXIT.
5669 030232 032737 000020 001564      BIT    #BIT4,SR2      ;SECOND A/D WITH WRAPARROUND MODULE?
5670 030240 001410                      BEQ    ETDF            ;NO-EXIT.
5671 030242 012737 030262 000004      MOV    #ETDF,#4       ;SET FOR TIMEOUT IF NOT ENOUGH CORE.
5672 030250 005037 000006                      CLR    #6
5673 030254 005737 070000      TST    #70000         ;ADDR. MEMORY.
5674 030260 000407                      BR     1$             ;GO TO START OF TEST,IF NOT ENOUGH MEMORY
5675
5676
5677
5678 030262 012737 000006 000004      ETDF=.  MOV    #6,#4           ;RESTORE LOC 4
5679 030270 012706 001100                      MOV    #STACK,SP
5680 030274 000137 030576                      JMP    20$
5681
5682 030300 012737 000006 000004      1$:    MOV    #6,#4
5683 030306 012737 000503 044114      MOV    #503,KWT+2     ;START CLOCK AT RATE 1MHZ.
5684 030314 012737 177763 044116      MOV    #-13.,KWT+4    ;INTR. EVERY 13 USEC.
5685 030322 012700 055066                      MOV    #JOB0,R0
5686 030326 012710 004442                      MOV    #BIT11!BITS!BITS!BIT1,(0) ;START WORD.
5687 030332 012760 001750 000002      MOV    #1000.,2(0)    ;BUFFER SIZE
5688 030340 012760 055164 000004      MOV    #JOB0U,4(0)    ;USW ARRD.
5689 030346 005070 000004                      CLR    #4(0)
5690 030352 012760 070000 000010      MOV    #70000,10(0)   ;SET UP BUFFER ADDRS.
5691 030360 012760 070000 000014      MOV    #70000,14(0)
5692 030366 062760 001750 000014      ADD    #1000.,14(0)
5693 030374 012760 070000 000020      MOV    #70000,20(0)
5694 030402 062760 003720 000020      ADD    #2000.,20(0)
5695 030410 112760 000002 000007      MOV    #2,7(0)
5696 030416 112760 000144 000054      MOV    #100.,54(0)   ;DELAY BEFORE START.
5697 030424 112760 000001 000060      MOV    #1.,60(0)     ;NUMBER OF CHANS.
5698 030432 112737 000002 034454      MOV    #2.,CHANU
5699 030440 012760 000001 000062      MOV    #1,62(0)      ;TICKS BETWEEN SAMPLES.
5700 030446 012760 000400 000056      MOV    #400,56(0)    ;START CHO,INC=1
5701 030454 012737 000001 001124      MOV    #1,$GDDAT     ;ONE JOB.
5702 030462 004737 036026                      JSR    PC,FLASH      ;DO IT.
5703
5704 030466 005237 034540                      INC    REPOR         ;LOOK AT AD11k #1
5705 030472 012703 055036                      MOV    #LIST12,R3
5706 030476 011337 034474      2$:    MOV    (3),CHANS
5707 030502 012337 034476      MOV    (3)+,CHANF
5708 030506 012337 034514      MOV    (3)+,AVEXP
5709 030512 012337 034512      MOV    (3)+,TOLER
5710 030516 012701 070000      MOV    #70000,R1
5711 030522 013737 034474 034466      MOV    CHANS,CHAN
5712 030530 004737 034160      JSR    PC,SPEP
5713
5714 030534 012703 055036                      MOV    #LIST12,R3   ;LOOK AT AD11k #2
5715 030540 011337 034474      MOV    (3),CHANS
5716 030544 012337 034476      MOV    (3)+,CHANF
5717 030550 012337 034514      MOV    (3)+,AVEXP
5718 030554 012337 034512      MOV    (3)+,TOLER
5719 030560 012701 070002      MOV    #70002,R1

```

```

5720 030564 013737 034474 034466      MOV      CHANS,CHAN
5721 030572 004737 034160      JSR      PC,SPEP
5722
5723 030576      20$:
5724
5725 030576      ETEST:
5726 *****
5727 *****
5728 *****
5729 030576 000004      *TEST S1      END OF TESTS
5730 *****
5731 *****
5732 *****
5733 *****
5734 *****
5735 *****
5736 *****
5737 030600      *INCREMENT THE PASS NUMBER ($PASS)
5738 030600 000240      *IF THERES A MONITOR GO TO IT
5739 030602 005037 001102      *IF THERE ISN'T JUMP TO RSTART
5740 030606 005037 001160
5741 030612 005237 001202
5742 030616 042737 100000 001202      $EOP:
5743 030624 005327      NOP
5744 030626 000001      CLR      $STNM      ;; ZERO THE TEST NUMBER
5745 030630 003064      CLR      $TIMES     ;; ZERO THE NUMBER OF ITERATIONS
5746 030632 012737      INC      $PASS      ;; INCREMENT THE PASS NUMBER
5747 030634 000001      BIC     #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
5748 030636 030626      DEC     (PC)+      ;; LOOP?
5749
5750
5751 030640 122737 000104 001772      $EOPCT: .WORD 1      ;; YES
5752 030646 001006      BGT     $DOAGN     ;; RESTORE COUNTER
5753 030650 104401 030656      MOV     (PC)+,a(PC)+
5754 030654 000402      $ENDCT: .WORD 1
5755
5756 030662      ;; /-ENDPAS-
5757 030662 000405      CMPB   #'D,$VERSN
5758 030664      BNE    1$
5759 030664 104401 030672      TYPE   65$      ;; TYPE ASCIZ STRING
5760 030670 000402      BR     64$      ;; GET OVER THE ASCIZ
5761
5762 030676      ;; 65$:
5763 030676      64$: BR     2$
5764 030676 104401 030704      1$: TYPE   67$      ;; TYPE ASCIZ STRING
5765 030702 000405      BR     66$      ;; GET OVER THE ASCIZ
5766
5767 030716      ;; 67$:
5768 030716 013746 001202      66$: .ASCIZ <200>#M-*
5769
5770 030722 104405      2$: TYPE   69$      ;; TYPE ASCIZ STRING
5771 030724 104401 030732      BR     68$      ;; GET OVER THE ASCIZ
5772 030730 000411      ;; 69$:
5773      68$: .ASCIZ #END PASS #
      MOV     $PASS,-(SP) ;; SAVE $PASS FOR TYPEOUT
      TYPDS  ;; TYPE PASS NUMBER.
      TYPE  71$      ;; GO TYPE--DECIMAL ASCII WITH SIGN
      BR     70$      ;; TYPE ASCIZ STRING
      ;; 71$: .ASCIZ # ; TOTAL ERRORS #

```

5774	030754				70\$:	MOV	ERCNT,-(SP)	::SAVE ERCNT FOR TYPEOUT
5775	030754	013746	001776			TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
5776	030760	104405			\$GET42:	MOV	@#42,R0	::GET MONITOR ADDRESS
5777	030762	013700	000042			BEQ	\$DOAgN	::BRANCH IF NO MONITOR
5778	030766	001405				RESET		::CLEAR THE WORLD
5779	030770	000005			\$ENDAD:	JSR	PC,(R0)	::GO TO MONITOR
5780	030772	004710				NOP		::SAVE ROOM
5781	030774	000240				NOP		::FOR
5782	030776	000240				NOP		::ACT11
5783	031000	000240			\$DOAgN:			
5784	031002					JMP	@(PC)+	::RETURN
5785	031002	000137			\$RTNAD:	.WORD	RSTART	
5786	031004	003500			\$ENULL:	.BYTE	-1,-1,0	::NULL CHARACTER STRING
5787	031006	377	377	000		.EVEN		
5788		031012						

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPDS  ;;CALL FOR TYPEOUT
*   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*   .BYTE  M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS

```

```

*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC

```

```

*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPON  ;;CALL FOR TYPEOUT

```

```

*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

```

```

*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOC  ;;CALL FOR TYPEOUT

```

5816	031012	017646	000000		\$TYPOS:	MOV	@(SP),-(SP)	::PICKUP THE MODE
5817	031016	116637	000001	031235		MOVB	1(SP),\$OFILL	::LOAD ZERO FILL SWITCH
5818	031024	112637	031237			MOVB	(SP)+,\$OMODE+1	::NUMBER OF DIGITS TO TYPE
5819	031030	062716	000002			ADD	#2,(SP)	::ADJUST RETURN ADDRESS
5820	031034	000406				BR	\$TYPON	
5821	031036	112737	000001	031235	\$TYPOC:	MOVB	#1,\$OFILL	::SET THE ZERO FILL SWITCH
5822	031044	112737	000006	031237		MOVB	#6,\$OMODE+1	::SET FOR SIX(6) DIGITS
5823	031052	112737	000005	031234	\$TYPON:	MOVB	#5,\$OCNT	::SET THE ITERATION COUNT
5824	031060	010346				MOV	R3,-(SP)	::SAVE R3
5825	031062	010446				MOV	R4,-(SP)	::SAVE R4
5826	031064	010546				MOV	R5,-(SP)	::SAVE R5
5827	031066	113704	031237			MOVB	\$OMODE+1,R4	::GET THE NUMBER OF DIGITS TO TYPE

```

5828 031072 005404          NEG      R4
5829 031074 062704 000006  ADD      #6,R4          ;; SUBTRACT IT FOR MAX. ALLOWED
5830 031100 110437 031236  MOVB    R4,$OMODE    ;; SAVE IT FOR USE
5831 031104 113704 031235  MOVB    $OFILL,R4    ;; GET THE ZERO FILL SWITCH
5832 031110 016605 000012  MOV     12(SP),R5    ;; PICKUP THE INPUT NUMBER
5833 031114 005003          CLR     R3           ;; CLEAR THE OUTPUT WORD
5834 031116 006105          1$:    ROL     R5           ;; ROTATE MSB INTO "C"
5835 031120 000404          BR     3$           ;; GO DO MSB
5836 031122 006105          2$:    ROL     R5           ;; FORM THIS DIGIT
5837 031124 006105          ROL     R5
5838 031126 006105          ROL     R5
5839 031130 010503          MOV     R5,R3
5840 031132 006103          3$:    ROL     R3           ;; GET LSB OF THIS DIGIT
5841 031134 105337 031236  DECB   $OMODE        ;; TYPE THIS DIGIT?
5842 031140 100016          BPL    7$           ;; BR IF NO
5843 031142 042703 177770  BIC    #177770,R3    ;; GET RID OF JUNK
5844 031146 001002          BNE    4$           ;; TEST FOR 0
5845 031150 005704          TST   R4           ;; SUPPRESS THIS 0?
5846 031152 001403          BEQ   5$           ;; BR IF YES
5847 031154 005204          4$:    INC    R4           ;; DON'T SUPPRESS ANYMORE 0'S
5848 031156 052703 000060  BIS    #'0,R3        ;; MAKE THIS DIGIT ASCII
5849 031162 052703 000040  5$:    BIS    #' ,R3    ;; MAKE ASCII IF NOT ALREADY
5850 031166 110337 031232  MOVB   R3,$$        ;; SAVE FOR TYPING
5851 031172 104401 031232  TYPE   8$           ;; GO TYPE THIS DIGIT
5852 031176 105337 031234  7$:    DECB   $OCNT    ;; COUNT BY 1
5853 031202 003347          BGT   2$           ;; BR IF MORE TO DO
5854 031204 002402          BLT   6$           ;; BR IF DONE
5855 031206 005204          INC   R4           ;; INSURE LAST DIGIT ISN'T A BLANK
5856 031210 000744          BR    2$           ;; GO DO THE LAST DIGIT
5857 031212 012605          6$:    MOV    (SP)+,R5    ;; RESTORE R5
5858 031214 012604          MOV    (SP)+,R4        ;; RESTORE R4
5859 031216 012603          MOV    (SP)+,R3        ;; RESTORE R3
5860 031220 016666 000002 000004  MOV    2(SP),4(SP)    ;; SET THE STACK FOR RETURNING
5861 031226 012616          MOV    (SP)+,(SP)
5862 031230 000002          RTI                    ;; RETURN
5863 031232 000          8$:    .BYTE 0          ;; STORAGE FOR ASCII DIGIT
5864 031233 000          .BYTE 0          ;; TERMINATOR FOR TYPE ROUTINE
5865 031234 000          $OCNT: .BYTE 0    ;; OCTAL DIGIT COUNTER
5866 031235 000          $OFILL: .BYTE 0   ;; ZERO FILL SWITCH
5867 031236 000000          $OMODE: .WORD 0   ;; NUMBER OF DIGITS TO TYPE
5868          .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
5869
5870          ;; *****
5871          ;; *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
5872          ;; *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
5873          ;; *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
5874          ;; *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
5875          ;; *REPLACED WITH SPACES.
5876          ;; *CALL:
5877          ;; *      MOV     NUM,-(SP)          ;; PUT THE BINARY NUMBER ON THE STACK
5878          ;; *      TYPDS          ;; GO TO THE ROUTINE
5879
5880          $TYPDS:
5881          MOV     RD,-(SP)          ;; PUSH RD ON STACK

```

```

5882 031242 010146      MOV      R1,-(SP)
5883 031244 010246      MOV      R2,-(SP)
5884 031246 010346      MOV      R3,-(SP)
5885 031250 010546      MOV      R5,-(SP)
5886 031252 012746 020200      MOV      #20200,-(SP)
5887 031256 016605 000020      MOV      20(SP),R5
5888 031262 100004      BPL      1$
5889 031264 005405      NEG      R5
5890 031266 112766 000055 000001      MOVVB   #'-',1(SP)
5891 031274 005000      CLR      R0
5892 031276 012703 031454      MOV      $DBLK,R3
5893 031302 112723 000040      MOVVB   #'',(R3)+
5894 031306 005002      CLR      R2
5895 031310 016001 031444      MOV      $DTBL(R0),R1
5896 031314 160105      SUB      R1,R5
5897 031316 002402      BLT     4$
5898 031320 005202      INC      R2
5899 031322 000774      BR      3$
5900 031324 060105      ADD      R1,R5
5901 031326 005702      TST     R2
5902 031330 001002      BNE     5$
5903 031332 105716      TSTB   (SP)
5904 031334 100407      BMI     7$
5905 031336 106316      ASLB   (SP)
5906 031340 103003      BCC     6$
5907 031342 116663 000001 177777      MOVVB   1(SP),-1(R3)
5908 031350 052702 000060      BIS     #'0,R2
5909 031354 052702 000040      BIS     #' ,R2
5910 031360 110223      MOVVB   R2,(R3)+
5911 031362 005720      TST     (R0)+
5912 031364 020027 000010      CMP     R0,#10
5913 031370 002746      BLT     2$
5914 031372 003002      BGT     8$
5915 031374 010502      MOV     R5,R2
5916 031376 000764      BR      6$
5917 031400 105726      TSTB   (SP)+
5918 031402 100003      BPL     9$
5919 031404 116663 177777 177776      MOVVB   -1(SP),-2(R3)
5920 031412 105013      CLRB   (R3)
5921 031414 012605      MOV     (SP)+,R5
5922 031416 012603      MOV     (SP)+,R3
5923 031420 012602      MOV     (SP)+,R2
5924 031422 012601      MOV     (SP)+,R1
5925 031424 C 2600      MOV     (SP)+,R0
5926 031426 104401 031454      TYPE    $DBLK
5927 031432 016666 000002 000004      MOV     2(SP),4(SP)
5928 031440 012616      MOV     (SP)+,(SP)
5929 031442 000002      RTI
5930 031444 023420      $DTBL: 10000.
5931 031446 001750      1000.
5932 031450 000144      100.
5933 031452 000012      10.
5934 031454 000004      $DBLK: .BLKW 4
5935                                     .SBTTL ERROR HANDLER ROUTINE

```

```

:; PUSH R1 ON STACK
:; PUSH R2 ON STACK
:; PUSH R3 ON STACK
:; PUSH R5 ON STACK
:; SET BLANK SWITCH AND SIGN
:; GET THE INPUT NUMBER
:; BR IF INPUT IS POS.
:; MAKE THE BINARY NUMBER POS.
:; MAKE THE ASCII NUMBER NEG.
:; ZERO THE CONSTANTS INDEX
:; SETUP THE OUTPUT POINTER
:; SET THE FIRST CHARACTER TO A BLANK
:; CLEAR THE BCD NUMBER
:; GET THE CONSTANT
:; FORM THIS BCD DIGIT
:; BR IF DONE
:; INCREASE THE BCD DIGIT BY 1
:; ADD BACK THE CONSTANT
:; CHECK IF BCD DIGIT=0
:; FALL THROUGH IF 0
:; STILL DOING LEADING 0'S?
:; BR IF YES
:; MSD?
:; BR IF NO
:; YES--SET THE SIGN
:; MAKE THE BCD DIGIT ASCII
:; MAKE IT A SPACE IF NOT ALREADY A DIGIT
:; PUT THIS CHARACTER IN THE OUTPUT BUFFER
:; JUST INCREMENTING
:; CHECK THE TABLE INDEX
:; GO DO THE NEXT DIGIT
:; GO TO EXIT
:; GET THE LSD
:; GO CHANGE TO ASCII
:; WAS THE LSD THE FIRST NON-ZERO?
:; BR IF NO
:; YES--SET THE SIGN FOR TYPING
:; SET THE TERMINATOR
:; POP STACK INTO R5
:; POP STACK INTO R3
:; POP STACK INTO R2
:; POP STACK INTO R1
:; POP STACK INTO R0
:; NOW TYPE THE NUMBER
:; ADJUST THE STACK
:; RETURN TO USER

```

```

5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948
5949 031464
5950 031464 104407
5951 031466 105237 001103
5952 031472 001775
5953 031474 013777 001102 147440
5954 031502 032777 002000 147430
5955 031510 001402
5956 031512 104401 001164
5957 031516 005237 001112
5958 031522 011637 001116
5959 031526 162737 000002 001116
5960 031534 117737 147356 001114
5961 031542 032777 020000 147370
5962 031550 001004
5963 031552 004737 031660
5964 031556 104401 001171
5965 031562
5966 031562 122737 000001 001214
5967 031570 001007
5968 031572 113737 001114 031604
5969 031600 004737 033454
5970 031604 000
5971 031605 000
5972 031606 000777
5973 031610 005777 147324
5974 031614 100002
5975 031616 000000
5976 031620 104407
5977 031622 032777 001000 147310
5978 031630 001402
5979 031632 013716 001110
5980 031636 005737 001162
5981 031642 001402
5982 031644 013716 001162
5983 031650
5984 031650 005237 001776
5985 031654 001775
5986 031656 000002
5987
5988
5989

```

```

;*****
;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
;AND GO TO SERrTYP ON ERROR
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;SW15=1 HALT ON ERROR
;SW13=1 INHIBIT ERROR TYPEOUT
;SW10=1 BELL ON ERROR
;SW09=1 LOOP ON ERROR
;CALL
;* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

$ERROR:
7$: CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
INCB $ERFLG ;; SET THE ERROR FLAG
BEQ 7$ ;; DON'T LET THE FLAG GO TO ZERO
MOV $STNM,@DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
BIT #BIT10,@SWR ;; BELL ON ERROR?
BEQ 1$ ;; NO - SKIP
TYPE $BELL ;; RING BELL
1$: INC $ERTTL ;; COUNT THE NUMBER OF ERRORS
MOV (SP), $ERRPC ;; GET ADDRESS OF ERROR INSTRUCTION
SUB #2, $ERRPC
MOVB @ $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13, @SWR ;; SKIP TYPEOUT IF SET
BNE 20$ ;; SKIP TYPEOUTS
JSR PC, $ERRTYP ;; GO TO USER ERROR ROUTINE
TYPE , $CRLF

20$: CMPB #APTENV, $ENV ;; RUNNING IN APT MODE
BNE 2$ ;; NO SKIP APT ERROR REPORT
MOVB $ITEMB, 21$ ;; SET ITEM NUMBER AS ERROR NUMBER
JSR PC, $ATY4 ;; REPORT FATAL ERROR TO APT

21$: .BYTE 0
.BYTE 0
22$: BR 22$ ;; APT ERROR LOOP
2$: TST @SWR ;; HALT ON ERROR
BPL 3$ ;; SKIP IF CONTINUE
HALT ;; HALT ON ERROR!
3$: CKSWR ;; TEST FOR CHANGE IN SOFT-SWR
BIT #BIT09, @SWR ;; LOOP ON ERROR SWITCH SET?
BEQ 4$ ;; BR IF NO
MOV $LPERR, (SP) ;; FUDGE RETURN FOR LOOPING
4$: TST $ESCAPE ;; CHECK FOR AN ESCAPE ADDRESS
BEQ 5$ ;; BR IF NONE
MOV $ESCAPE, (SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE

5$: 10$: INC ERCNT
BEQ 10$
RTI
.SBTTL ERROR MESSAGE TYPEOUT ROUTINE
;*****

```



```

5990 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
5991 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
5992 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
5993
5994 031660 $ERRTYP:
5995 031660 104401 001171 TYPE $CRLF ;;"CARRIAGE RETURN" & "LINE FEED"
5996 031664 010046 MOV RO,-(SP) ;:SAVE RO
5997 031666 005000 CLR RO ;:PICKUP THE ITEM INDEX
5998 031670 153700 001114 BISB @*$ITEMB,RO
5999 031674 001004 BNE 1$ ;:IF ITEM NUMBER IS ZERO, JUST
6000 ;:TYPE THE PC OF THE ERROR
6001 031676 013746 001116 MOV $ERRPC,-(SP) ;:SAVE $ERRPC FOR TYPEOUT
6002 ;:ERROR ADDRESS
6003 031702 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
6004 031704 000426 BR 6$ ;:GET OUT
6005 031706 005300 1$: DEC RO ;:ADJUST THE INDEX SO THAT IT WILL
6006 031710 006300 ASL RO ;:WORK FOR THE ERROR TABLE
6007 031712 006300 ASL RO
6008 031714 006300 ASL RO
6009 031716 062700 001256 ADD #$ERRTB,RO ;:FORM TABLE POINTER
6010 031722 012037 031732 MOV (RO)+,2$ ;:PICKUP "ERROR MESSAGE" POINTER
6011 031726 001404 BEQ 3$ ;:SKIP TYPEOUT IF NO POINTER
6012 031730 104401 TYPE ;:TYPE THE "ERROR MESSAGE"
6013 031732 000000 2$: .WORD 0 ;:"ERROR MESSAGE" POINTER GOES HERE
6014 031734 104401 001171 TYPE $CRLF ;;"CARRIAGE RETURN" & "LINE FEED"
6015 031740 012037 031750 3$: MOV (RO)+,4$ ;:PICKUP "DATA HEADER" POINTER
6016 031744 001404 BEQ 5$ ;:SKIP TYPEOUT IF 0
6017 031746 104401 TYPE ;:TYPE THE "DATA HEADER"
6018 031750 000000 4$: .WORD 0 ;:"DATA HEADER" POINTER GOES HERE
6019 031752 104401 001171 TYPE $CRLF ;;"CARRIAGE RETURN" & "LINE FEED"
6020 031756 011000 5$: MOV (RO),RO ;:PICKUP "DATA TABLE" POINTER
6021 031760 001004 BNE 7$ ;:GO TYPE THE DATA
6022 031762 012600 6$: MOV (SP)+,RO ;:RESTORE RO
6023 031764 104401 001171 TYPE $CRLF ;;"CARRIAGE RETURN" & "LINE FEED"
6024 031770 000207 RTS PC ;:RETURN
6025 031772
6026 031772 013046 7$: MOV @ (RO)+,-(SP) ;:SAVE @ (RO)+ FOR TYPEOUT
6027 031774 104402 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
6028 031776 005710 TST (RO) ;:IS THERE ANOTHER NUMBER?
6029 032000 001770 BEQ 6$ ;:BR IF NO
6030 032002 104401 032010 TYPE 8$ ;:TYPE TWO(2) SPACES
6031 032006 000771 BR 7$ ;:LOOP
6032 032010 020040 000 8$: .ASCIZ / / ;:TWO(2) SPACES
6033 032014 .EVEN
6034 .SBTTL SCOPE HANDLER ROUTINE
6035
6036 ;*****
6037 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
6038 ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
6039 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
6040 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6041 ;*SW14=1 LOOP ON TEST
6042 ;*SW11=1 INHIBIT ITERATIONS
6043 ;*SW09=1 LOOP ON ERROR
    
```

```

6044 ;*SW08=1          LOOP ON TEST IN SWR<7:0>
6045 ;*CALL
6046 ;*          SCOPE          ;;SCOPE=IOT
6047
6048 $SCOPE:
6049          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
6050          CKSWR
6051 1$:          BIT          #BIT14,2SWR          ;;LOOP ON PRESENT TEST?
6052          BNE          $OVER          ;;YES IF SW14=1
6053          ;*****START OF CODE FOR THE XOR TESTER*****
6054 $XTSTR: BR          6$          ;;IF RUNNING ON THE "XOR" TESTER CHANGE
6055          ;THIS INSTRUCTION TO A "NOP" (NOP=240)
6056          MOV          2#ERRVEC, -(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
6057          MOV          #55, 2#ERRVEC          ;;SET FOR TIMEOUT
6058          TST          2#177060          ;;TIME OUT ON XOR?
6059          MOV          (SP)+, 2#ERRVEC          ;;RESTORE THE ERROR VECTOR
6060          BR          $SVLAD          ;;GO TO THE NEXT TEST
6061          5$:          CMP          (SP)+, (SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
6062          MOV          (SP)+, 2#ERRVEC          ;;RESTORE THE ERROR VECTOR
6063          BR          7$          ;;LOOP ON THE PRESENT TEST
6064          6$: ;*****END OF CODE FOR THE XOR TESTER*****
6065          BIT          #BIT08, 2SWR          ;;LOOP ON SPEC. TEST?
6066          BEQ          2$          ;;BR IF NO
6067          CMPB         2SWR, $STSNM          ;;ON THE RIGHT TEST? SWR<7:0>
6068          BEQ          $OVER          ;;BR IF YES
6069          2$:          TSTB         $ERFLG          ;;HAS AN ERROR OCCURRED?
6070          BEQ          3$          ;;BR IF NO
6071          CMPB         $ERMAX, $ERFLG          ;;MAX. ERRORS FOR THIS TEST OCCURRED?
6072          BHI          3$          ;;BR IF NO
6073          BIT          #BIT09, 2SWR          ;;LOOP ON ERROR?
6074          BEQ          4$          ;;BR IF NO
6075          7$:          MOV          $LPERR, $LPADR          ;;SET LOOP ADDRESS TO LAST SCOPE
6076          BR          $OVER
6077          4$:          CLRB         $ERFLG          ;;ZERO THE ERROR FLAG
6078          CLR          $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
6079          BR          1$          ;;ESCAPE TO THE NEXT TEST
6080          3$:          BIT          #BIT11, 2SWR          ;;INHIBIT ITERATIONS?
6081          BNE          1$          ;;BR IF YES
6082          TST          $PASS          ;;IF FIRST PASS OF PROGRAM
6083          BEQ          1$          ;;INHIBIT ITERATIONS
6084          INC          $ICNT          ;;INCREMENT ITERATION COUNT
6085          CMP          $TIMES, $ICNT          ;;CHECK THE NUMBER OF ITERATIONS MADE
6086          BGE          $OVER          ;;BR IF MORE ITERATION REQUIRED
6087          1$:          MOV          #1, $ICNT          ;;REINITIALIZE THE ITERATION COUNTER
6088          MOV          $MXCNT, $TIMES          ;;SET NUMBER OF ITERATIONS TO DO
6089          $SVLAD:      INCB         $STSNM          ;;COUNT TEST NUMBERS
6090          MOVB         $STSNM, $TESTN          ;;SET TEST NUMBER IN APT MAILBOX
6091          MOV          (SP), $LPADR          ;;SAVE SCOPE LOOP ADDRESS
6092          MOV          (SP), $LPERR          ;;SAVE ERROR LOOP ADDRESS
6093          CLR          $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
6094          MOVB         #1, $ERMAX          ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
6095          $OVER:      MOV          $STSNM, 2$DISPLAY          ;;DISPLAY TEST NUMBER
6096          MOV          $LPADR, (SP)          ;;FUDGE RETURN ADDRESS
6097          RTI          ;;FIXES PS

```

```

6098 032274 000010          SMXCNT: 10          ;;MAX. NUMBER OF ITERATIONS
6099                      SBTTL  TTY INPUT ROUTINE
6100
6101                      ;:*****
6102                      ;ENABL  LSB
6103
6104                      ;:*****
6105                      ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
6106                      ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
6107                      ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
6108                      ;*WHEN OPERATING IN TTY FLAG MODE.
6109 032276 022737 000176 001140 $CKSWR: CMP      #SWREG, SWR          ;; IS THE SOFT-SWR SELECTED?
6110 032304 001074          BNE      15$          ;; BRANCH IF NO
6111 032306 105777 146632          TSTB    @STKS          ;; CHAR THERE?
6112 032312 100071          BPL      15$          ;; IF NO, DON'T WAIT AROUND
6113 032314 117746 146626          MOVB    @STKB, -(SP)      ;; SAVE THE CHAR
6114 032320 042716 177600          BIC     #↑C17?, (SP)    ;; STRIP-OFF THE ASCII
6115 032324 022726 000007          CMP     #7, (SP)+      ;; IS IT A CONTROL G?
6116 032330 001062          BNE      15$          ;; NO, RETURN TO USER
6117 032332 123727 001134 000001  CMPB    $AUTOB, #1      ;; ARE WE RUNNING IN AUTO-MODE?
6118 032340 001456          BEQ     15$          ;; BRANCH IF YES
6119
6120 032342 104401 033023          TYPE   , $CNTLG          ;; ECHO THE CONTROL-G (↑G)
6121 032346 104401 033030          TYPE   , $MSWR          ;; TYPE CURRENT CONTENTS
6122 032352 013746 000176          MOV     SWREG, -(SP)    ;; SAVE SWREG FOR TYPEOUT
6123 032356 104402          TYPOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
6124 032360 104401 033041          TYPE   , $MNEW          ;; PROMPT FOR NEW SWR
6125 032364 005046          CLR     -(SP)          ;; CLEAR COUNTER
6126 032366 005046          CLR     -(SP)          ;; THE NEW SWR
6127 032370 105777 146550          TSTB    @STKS          ;; CHAR THERE?
6128 032374 100775          BPL     7$            ;; IF NOT TRY AGAIN
6129
6130 032376 117746 146544          MOVB    @STKB, -(SP)    ;; PICK UP CHAR
6131 032402 042716 177600          BIC     #↑C17?, (SP)    ;; MAKE IT 7-BIT ASCII
6132
6133
6134
6135 032406 021627 000025          9$:    CMP     (SP), #25      ;; IS IT A CONTROL-U?
6136 032412 001005          BNE     10$          ;; BRANCH IF NOT
6137 032414 104401 033016          TYPE   , $CNTLU          ;; YES, ECHO CONTROL-U (↑U)
6138 032420 062706 000006          20$:   ADD     #6, SP          ;; IGNORE PREVIOUS INPUT
6139 032424 000757          BR      19$          ;; LET'S TRY IT AGAIN
6140
6141
6142 032426 021627 000015          10$:   CMP     (SP), #15      ;; IS IT A <CR>?
6143 032432 001022          BNE     16$          ;; BRANCH IF NO
6144 032434 005766 000004          TST     4(SP)          ;; YES, IS IT THE FIRST CHAR?
6145 032440 001403          BEQ     11$          ;; BRANCH IF YES
6146 032442 016677 000002 146470          MOV     2(SP), @SWR      ;; SAVE NEW SWR
6147 032450 062706 000006          ADD     #6, SP          ;; CLEAR UP STACK
6148 032454 104401 001171          11$:   TYPE   , $CRLF          ;; ECHO <CR> AND <LF>
6149 032460 123727 001135 000001  14$:   CMPB    $INTAG, #1      ;; RE-ENABLE TTY KBD INTERRUPTS?
6150 032466 001003          BNE     15$          ;; BRANCH IF NOT
6151 032470 012777 000100 146446          MOV     #100, @STKS     ;; RE-ENABLE TTY KBD INTERRUPTS

```

```

6152 032476 000002
6153 032500 004737 033264
6154 032504 021627 000060
6155 032510 002420
6156 032512 021627 000067
6157 032516 003015
6158 032520 042726 000060
6159 032524 005766 000002
6160 032530 001403
6161 032532 006316
6162 032534 006316
6163 032536 006316
6164 032540 005266 000002
6165 032544 056616 177776
6166 032550 000707
6167 032552 104401 001170
6168 032556 000720
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180 032560 011646
6181 032562 016666 000004 000002
6182 032570 105777 146350
6183 032574 100375
6184 032576 117766 146344 000004
6185 032604 042766 177600 000004
6186 032612 026627 000004 000023
6187 032620 001013
6188 032622 105777 146316
6189 032626 100375
6190 032630 117746 146312
6191 032634 042716 177600
6192 032640 022627 000021
6193 032644 001366
6194 032646 000750
6195 032650 026627 000004 000140
6196 032656 002407
6197 032660 026627 000004 000175
6198 032666 003003
6199 032670 042766 000040 000004
6200 032676 000002
6201
6202
6203
6204
6205

```

```

15$: RTI
16$: JSR PC,$TYPEC
CMP (SP),#60
BLT 18$
CMP (SP),#67
BGT 18$
BIC #60,(SP)+
TST 2(SP)
BEQ 17$
ASL (SP)
ASL (SP)
ASL (SP)
17$: INC 2(SP)
BIS -2(SP),(SP)
BR 7$
18$: TYPE $QUES
BR 20$
.DSABL LSB

```

```

: RETURN
: ECHO CHAR
: CHAR < 0?
: BRANCH IF YES
: CHAR > 7?
: BRANCH IF YES
: STRIP-OFF ASCII
: IS THIS THE FIRST CHAR
: BRANCH IF YES
: NO, SHIFT PRESENT
: CHAR OVER TO MAKE
: ROOM FOR NEW ONE.
: KEEP COUNT OF CHAR
: SET IN NEW CHAR
: GET THE NEXT ONE
: TYPE ?<CR><LF>
: SIMULATE CONTROL-U

```

```

:*****
:THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

```

```

:CALL:
:* RDCHR : INPUT A SINGLE CHARACTER FROM THE TTY
:* RETURN HERE : CHARACTER IS ON THE STACK
:* : WITH PARITY BIT STRIPPED OFF

```

```

$RDCHR: MOV (SP),-(SP) : PUSH DOWN THE PC
MOV 4(SP),2(SP) : SAVE THE PS
1$: TSTB @STKS : WAIT FOR
BPL 1$ : A CHARACTER
MOVB @STKB,4(SP) : READ THE TTY
BIC #177,4(SP) : GET RID OF JUNK IF ANY
CMP 4(SP),#23 : IS IT A CONTROL-S?
BNE 3$ : BRANCH IF NO
2$: TSTB @STKS : WAIT FOR A CHARACTER
BPL 2$ : LOOP UNTIL ITS THERE
MOVB @STKB,-(SP) : GET CHARACTER
BIC #177,(SP) : MAKE IT 7-BIT ASCII
CMP (SP)+,#21 : IS IT A CONTROL-Q?
BNE 2$ : IF NOT DISCARD IT
BR 1$ : YES, RESUME
3$: CMP 4(SP),#140 : IS IT UPPER CASE?
BLT 4$ : BRANCH IF YES
CMP 4(SP),#175 : IS IT A SPECIAL CHAR?
BGT 4$ : BRANCH IF YES
BIC #40,4(SP) : MAKE IT UPPER CASE
4$: RTI : GO BACK TO USER

```

```

:*****
:THIS ROUTINE WILL INPUT A STRING FROM THE TTY

```

```

:CALL:
:* RDLIN : INPUT A STRING FROM THE TTY
:* RETURN HERE : ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK

```

```

6206 ;* ;: TERMINATOR WILL BE A BYTE OF ALL 0'S
6207
6208 032700 010346 $RDLIN: MOV R3 -(SP) ;: SAVE R3
6209 032702 012703 033006 1$: MOV #STTYIN,R3 ;: GET ADDRESS
6210 032706 022703 033016 2$: CMP #STTYIN+8.,R3 ;: BUFFER FULL?
6211 032712 101405 BLOS 4$ ;: BR IF YES
6212 032714 104410 RDCHR ;: GO READ ONE CHARACTER FROM THE TTY
6213 032716 112613 MOV (SP)+,(R3) ;: GET CHARACTER
6214 032720 122713 000177 10$: CMPB #177,(R3) ;: IS IT A RUBOUT
6215 032724 001003 BNE 3$ ;: SKIP IF NOT
6216 032726 104401 001170 4$: TYPE $QUES ;: TYPE A '?'
6217 032732 000763 BR 1$ ;: CLEAR THE BUFFER AND LOOP
6218 032734 111337 033004 3$: MOV (R3),9$ ;: ECHO THE CHARACTER
6219 032740 104401 033004 TYPE 9$
6220 032744 122723 000015 CMPB #15,(R3)+ ;: CHECK FOR RETURN
6221 032750 001356 BNE 2$ ;: LOOP IF NOT RETURN
6222 032752 105063 177777 CLRB -1(R3) ;: CLEAR RETURN (THE 15)
6223 032756 104401 001172 TYPE $LF ;: TYPE A LINE FEED
6224 032762 012603 MOV (SP)+,R3 ;: RESTORE R3
6225 032764 011646 MOV (SP)-,(SP) ;: ADJUST THE STACK AND PUT ADDRESS OF THE
6226 032766 016666 000004 000002 MOV 4(SP),2(SP) ;: FIRST ASCII CHARACTER ON IT
6227 032774 012766 033006 000004 MOV #STTYIN,4(SP)
6228 033002 000002 RTI ;: RETURN
6229 033004 000 9$: .BYTE 0 ;: STORAGE FOR ASCII CHAR. TO TYPE
6230 033005 000 .BYTE 0 ;: TERMINATOR
6231 033006 000010 .BLKB 8. ;: RESERVE 8 BYTES FOR TTY INPUT
6232 033016 052536 005015 000 $CNTLU: .ASCIZ /↑U/<15><12> ;: CONTROL "U"
6233 033023 136 006507 000012 $CNTLG: .ASCIZ /↑G/<15><12> ;: CONTROL "G"
6234 033030 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
6235 033036 020075 000 $MNEW: .ASCIZ / NEW = /
6236 033041 040 047040 053505
6237 033046 036440 000040
6238 ; .SSB2D
6239 ; .SOB2D
6240
6241 .SBTTL TYPE ROUTINE
6242
6243 ;: *****
6244 ;: *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
6245 ;: *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
6246 ;: *NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
6247 ;: *NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
6248 ;: *NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
6249 ;:
6250 ;: *CALL:
6251 ;: *1) USING A TRAP INSTRUCTION
6252 ;: * TYPE ,MESADR ;: MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
6253 ;: *OR
6254 ;: * TYPE
6255 ;: * MESADR
6256 ;: *
6257
6258 033052 105737 001157 $TYPE: TSTB $TPFLG ;: IS THERE A TERMINAL?
6259 033056 100002 BPL 1$ ;: BR IF YES

```

6260	033060	000000			HALT		;; HALT HERE IF NO TERMINAL
6261	033062	000430			BR	3\$;; LEAVE
6262	033064	010046			MOV	RO, -(SP)	;; SAVE RO
6263	033066	017600	000002		MOV	2(SP), RO	;; GET ADDRESS OF ASCIZ STRING
6264	033072	122737	000001	001214	CMPB	#APTENV, \$ENV	;; RUNNING IN APT MODE
6265	033100	001011			BNE	62\$;; NO GO CHECK FOR APT CONSOLE
6266	033102	132737	000100	001215	BITB	#AP-SPOOL, \$ENVm	;; SPOOL MESSAGE TO APT
6267	033110	001405			BEQ	62\$;; NO GO CHECK FOR CONSOLE
6268	033112	010037	033122		MOV	RO, 61\$;; SETUP MESSAGE ADDRESS FOR APT
6269	033116	004737	033444		JSR	PC, \$ATY3	;; SPOOL MESSAGE TO APT
6270	033122	000000			.WORD	0	;; MESSAGE ADDRESS
6271	033124	132737	000040	001215	BITB	#APTCSUP, \$ENVm	;; APT CONSOLE SUPPRESSED
6272	033132	001003			BNE	60\$;; YES, SKIP TYPE OUT
6273	033134	112046			MOVB	(RO)+, -(SP)	;; PUSH CHARACTER TO BE TYPED UNTO STACK
6274	033136	001005			BNE	4\$;; BR IF IT ISN'T THE TERMINATOR
6275	033140	005726			TST	(SP)+	;; IF TERMINATOR POP IT OFF THE STACK
6276	033142	012600			MOV	(SP)+, RO	;; RESTORE RO
6277	033144	062716	000002		ADD	#2, (SP)	;; ADJUST RETURN PC
6278	033150	000002			RTI		;; RETURN
6279	033152	122716	000011		CMPB	#HT, (SP)	;; BRANCH IF <HT>
6280	033156	001430			BEQ	8\$	
6281	033160	122716	000200		CMPB	#CRLF, (SP)	;; BRANCH IF NOT <CRLF>
6282	033164	001006			BNE	5\$	
6283	033166	005726			TST	(SP)+	;; POP <CR><LF> EQUIV
6284	033170	104401			TYPE		;; TYPE A CR AND LF
6285	033172	001171			\$CRLF		
6286	033174	105037	033330		CLRB	\$CHARCNT	;; CLEAR CHARACTER COUNT
6287	033200	000755			BR	2\$;; GET NEXT CHARACTER
6288	033202	004737	033264		JSR	PC, \$TYPEC	;; GO TYPE THIS CHARACTER
6289	033206	123726	001156		CMPB	\$FILLC, (SP)+	;; IS IT TIME FOR FILLER CHARS.?
6290	033212	001350			BNE	2\$;; IF NO GO GET NEXT CHAR.
6291	033214	013746	001154		MOV	\$NULL, -(SP)	;; GET # OF FILLER CHARS. NEEDED
6292							;; AND THE NULL CHAR.
6293	033220	105366	000001		DECB	1(SP)	;; DOES A NULL NEED TO BE TYPED?
6294	033224	002770			BLT	6\$;; BR IF NO--GO POP THE NULL OFF OF STACK
6295	033226	004737	033264		JSR	PC, \$TYPEC	;; GO TYPE A NULL
6296	033232	105337	033330		DECB	\$CHARCNT	;; DO NOT COUNT AS A COUNT
6297	033236	000770			BR	7\$;; LOOP
6298							
6299							
6300							
6301	033240	112716	000040		MOVB	#' (SP)	;; REPLACE TAB WITH SPACE
6302	033244	004737	033264		JSR	PC, \$TYPEC	;; TYPE A SPACE
6303	033250	132737	000007	033330	BITB	#7, \$CHARCNT	;; BRANCH IF NOT AT
6304	033256	001372			BNE	9\$;; TAB STOP
6305	033260	005726			TST	(SP)+	;; POP SPACE OFF STACK
6306	033262	000724			BR	2\$;; GET NEXT CHARACTER
6307	033264	105777	145660		TSTB	2\$TPS	;; WAIT UNTIL PRINTER IS READY
6308	033270	100375			BPL	\$TYPEC	
6309	033272	116677	000002	145652	MOVB	2(SP), 2\$TPB	;; LOAD CHAR TO BE TYPED INTO DATA REG.
6310	033300	122766	000015	000002	CMPB	#CR, 2(SP)	;; IS CHARACTER A CARRIAGE RETURN?
6311	033306	001003			BNE	1\$;; BRANCH IF NO
6312	033310	105037	033330		CLRB	\$CHARCNT	;; YES--CLEAR CHARACTER COUNT
6313	033314	000406			BR	\$TYPEX	;; EXIT

; HORIZONTAL TAB PROCESSOR

```

6314 033316 122766 000012 000002 1$: CMPB #LF,2(SP) ;; IS CHARACTER A LINE FEED?
6315 033324 001,02 BEQ $TYPEX ;; BRANCH IF YES
6316 033326 105227 INCB (PC)+ ;; COUNT THE CHARACTER
6317 033330 000000 $CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
6318 033332 000207 $TYPEX: RTS PC
6319
6320 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
6321
6322 ;*****
6323 ;*THIS ROUTINE WILL READ AN OCTA (ASCII) NUMBER FROM THE TTY AND
6324 ;*CHANGE IT TO BINARY.
6325 ;*CALL:
6326 ;* RDOCT ;; READ AN OCTAL NUMBER
6327 ;* RETURN HERE ;; LOW ORDER BITS ARE ON TOP OF THE STACK
6328 ;* ;; HIGH ORDER BITS ARE IN $HIOCT
6329
6330 033334 011646 $RDOCT: MOV (SP),-(SP) ;; PROVIDE SPACE FOR THE
6331 033336 016666 000004 000002 MOV 4(SP),2(SP) ;; INPUT NUMBER
6332 033344 010046 MOV RO,-(SP) ;; PUSH RO ON STACK
6333 033346 010146 MOV R1,-(SP) ;; PUSH R1 ON STACK
6334 033350 010246 MOV R2,-(SP) ;; PUSH R2 ON STACK
6335 033352 104411 1$: RDLIN ;; READ AN ASCII LINE
6336 033354 012600 MOV (SP)+,RO ;; GET ADDRESS OF 1ST CHARACTER
6337 033356 005001 CLR R1 ;; CLEAR DATA WORD
6338 033360 005002 CLR R2
6339 033362 112046 2$: MOVB (RO)+,-(SP) ;; PICKUP THIS CHARACTER
6340 033364 001412 BEQ 3$ ;; IF ZERO GET OUT
6341 033366 006301 ASL R1 ;; *2
6342 033370 006102 ROL R2
6343 033372 006301 ASL R1 ;; *4
6344 033374 006102 ROL R2
6345 033376 006301 ASL R1 ;; *8
6346 033400 006102 ROL R2
6347 033402 042716 177770 BIC #1C7,(SP) ;; STRIP THE ASCII JUNK
6348 033406 062601 ADD (SP)+,R1 ;; ADD IN THIS DIGIT
6349 033410 000764 BR 2$ ;; LOOP
6350 033412 005726 3$: TST (SP)+ ;; CLEAN TERMINATOR FROM STACK
6351 033414 010166 000012 MOV R1,12(SP) ;; SAVE THE RESULT
6352 033420 010237 033434 MOV R2,$HIOCT
6353 033424 012602 MOV (SP)+,R2 ;; POP STACK INTO R2
6354 033426 012601 MOV (SP)+,R1 ;; POP STACK INTO R1
6355 033430 012600 MOV (SP)+,RO ;; POP STACK INTO RO
6356 033432 000002 RTI ;; RETURN
6357 033434 000000 $HIOCT: .WORD 0 ;; HIGH ORDER BITS GO HERE
6358
6359 .SBTTL APT COMMUNICATIONS ROUTINE
6360
6361 033436 112737 000001 033702 ;*****
6362 033444 112737 000001 033700 $ATY1: MOVB #1,$FFLG ;; TO REPORT FATAL ERROR
6363 033452 000403 $ATY3: MOVB #1,$MFLG ;; TO TYPE A MESSAGE
6364 033454 112737 000001 033702 $ATY4: BR $ATYC
6365 033462 $ATYC: MOVB #1,$FFLG ;; TO ONLY REPORT FATAL ERROR
6366 033462 010046 MOV RO,-(SP) ;; PUSH RO ON STACK
6367 033464 010146 MOV R1,-(SP) ;; PUSH R1 ON STACK

```

```

6368 033466 105737 033700          TSTB    $MFLG          ;; SHOULD TYPE A MESSAGE?
6369 033472 001450          BEQ     5$            ;; IF NOT: BR
6370 033474 122737 000001 001214    CMPB    #APTENV,$ENV  ;; OPERATING UNDER APT?
6371 033502 001031          BNE     3$            ;; IF NOT: BR
6372 033504 132737 000100 001215    BITB    #APTPOOL,$ENVM ;; SHOULD SPOOL MESSAGES?
6373 033512 001425          BEQ     3$            ;; IF NOT: BR
6374 033514 017600 000004          MOV     24(SP),RO    ;; GET MESSAGE ADDR.
6375 033520 062766 000002 000004    ADD     #2,4(SP)     ;; BUMP RETURN ADDR.
6376 033526 005737 001174          1$:    TST     $MSGTYPE   ;; SEE IF DONE W/ LAST XMISSION?
6377 033532 001375          BNE     1$            ;; IF NOT: WAIT
6378 033534 010037 001210          MOV     RO,$MSGAD    ;; PUT ADDR IN MAILBOX
6379 033540 105720          2$:    TSTB    (RO)+     ;; FIND END OF MESSAGE
6380 033542 001376          BNE     2$
6381 033544 163700 001210          SUB     $MSGAD,RO    ;; SUB START OF MESSAGE
6382 033550 006200          ASR     RO           ;; GET MESSAGE LNTH IN WORDS
6383 033552 010037 001212          MOV     RO,$MSGLEN   ;; PUT LENGTH IN MAILBOX
6384 033556 012737 000004 001174    MOV     #4,$MSGTYPE  ;; TELL APT TO TAKE MSG.
6385 033564 000413          BR      5$
6386 033566 017637 000004 033612 3$:    MOV     24(SP),4$    ;; PUT MSG ADDR IN JSR LINKAGE
6387 033574 062766 000002 000004    ADD     #2,4(SP)     ;; BUMP RETURN ADDRESS
6388 033602 013746 177776          MOV     177776,-(SP) ;; PUSH 177776 ON STACK
6389 033606 004737 033052          JSR     PC,$TYPE     ;; CALL TYPE MACRO
6390 033612 000000          4$:    .WORD    0
6391 033614          5$:
6392 033614 105737 033702          10$:   TSTB    $FFLG       ;; SHOULD REPORT FATAL ERROR?
6393 033620 001416          BEQ     12$          ;; IF NOT: BR
6394 033622 005737 001214          TST     $ENV         ;; RUNNING UNDER APT?
6395 033626 001413          BEQ     12$          ;; IF NOT: BR
6396 033630 005737 001174          11$:   TST     $MSGTYPE   ;; FINISHED LAST MESSAGE?
6397 033634 001375          BNE     11$          ;; IF NOT: WAIT
6398 033636 017637 000004 001176    MOV     24(SP),$FATAL ;; GET ERROR #
6399 033644 062766 000002 000004    ADD     #2,4(SP)     ;; BUMP RETURN ADDR.
6400 033652 005237 001174          INC     $MSGTYPE    ;; TELL APT TO TAKE ERROR
6401 033656 105037 033702          12$:   CLRB    $FFLG     ;; CLEAR FATAL FLAG
6402 033662 105037 033701          CLRB    $LFLG       ;; CLEAR LOG FLAG
6403 033666 105037 033700          CLRB    $MFLG       ;; CLEAR MESSAGE FLAG
6404 033672 012601          MOV     (SP)+,R1    ;; POP STACK INTO R1
6405 033674 012600          MOV     (SP)+,RO    ;; POP STACK INTO RO
6406 033676 000207          RTS     PC          ;; RETURN
6407 033700          $MFLG: .BYTE    0   ;; MESSG. FLAG
6408 033701          $LFLG: .BYTE    0   ;; LOG FLAG
6409 033702          $FFLG: .BYTE    0   ;; FATAL FLAG
6410          .EVEN
6411          APTSIZE=200
6412          APTENV=001
6413          APTPOOL=100
6414          APTCSUP=040
6415          .SBTTL  POWER DOWN AND UP ROUTINES
6416
6417          ;:*****
6418          ;:POWER DOWN ROUTINE
6419 033704 012737 034044 000024  $PWRDN: MOV     #5ILLUP,2#PWRVEC ;; SET FOR FAST UP
6420 033712 012737 000340 000026    MOV     #340,2#PWRVEC+2 ;; PRIO:7
6421 033720 010046          MOV     RO,-(SP)    ;; PUSH RO ON STACK

```


MAINDEC -11- DRLPA-A
DRLPA.P11 ; WIR

MACY11 27(654) 13-DEC-77 12:58 PAGE 139
ROUTINE TO WAIT FOR "DRLPX0" MICROCODE TO BECOME READY.

SEQ 0169

6465 034102 001371
6466
6467 034104 104003
6468
6469 034106

BNE 1\$

ERROR 3

;TIME OUT ERROR,KMC-11 ERROR.

2\$:

```

6470 034106 000207
6471 034110 000000
6472
6473
6474
6475
6476
6477
6478
6479
6480
6481 034112 004737 040626 SYNC: JSR PC,SRESET ;RESET SYSTEM
6482
6483 034116 012777 044112 145352 MOV #KWT,ALPADL ;CLOCK INFO
6484 034124 052777 000001 145334 BIS #BIT0,ALPCI ;START
6485 034132 004737 040542 JSR PC,SDLAY ;DELA
6486 034136 117737 145332 001126 MOVB ALPS0,$BDDAT ;DUMB READ OF STAT REG.
6487 034144 000207 RTS PC
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500
6501
6502
6503 034146 013737 034474 034466 AVERR: MOV CHANS,CHAN ;SET TO FIRST CHAN.
6504 034154 012701 056246 MOV #BUFFER,R1
6505 034160 013737 034454 034510 SPEP: MOV CHANU,SAMOFF
6506 034166 006337 034510 ASL SAMOFF
6507 034172 013737 034466 034464 MOV CHAN,CHAN1
6508 034200 006337 034464 ASL CHAN1
6509 034204 012737 000020 034506 AVERRL: MOV #16,SAMCNT
6510 034212 063701 034464 ADD CHAN1,R1 ;PUT CHAN OFFSET IN R1.
6511 034216 005037 034516 CLR AVTKN ;SET INITIAL CONDITIONS FOR THIS CHAN.
6512 034222 005037 034524 CLR RTEMP
6513 034226 011137 034520 MOV (1),RLOW
6514 034232 011137 034522 MOV (1),RHIGH
6515 034236 023711 034522 2$: CMP RHIGH,(1) ;FIND REAL HIGH VALUE.
6516 034242 003002 BGT 3$
6517 034244 011137 034522 MOV (1),RHIGH
6518 034250 021137 034520 3$: CMP (1),RLOW ;FIND REAL LOW VALUE.
6519 034254 003002 BGT 4$
6520 034256 011137 034520 MOV (1),RLOW
6521 034262 011137 034526 4$: MOV (1),RTEMP1 ;GET CURRENT SAMPLE.
6522 034266 063737 034526 034516 ADD RTEMP1,AVTKN ;"BOOT" ADD ALL SAMPLES.
6523 034274 005537 034524 ADC RTEMP

```

```

6524 034300 063701 034510 ADD SAMOFF,R1
6525 034304 005337 034506 DEC SAMCNT
6526 034310 001352 BNE 2$
6527 034312 013737 034514 034526 MOV AVEXP,RTEMP1
6528 034320 063737 034512 034526 ADD TOLER,RTEMP1
6529 034326 023737 034526 034522 CMP RTEMP1,RHIGH
6530 034334 002427 BLT ERAV1 ;NO-THEN REPORT ERROR.
6531 034336 163737 034512 034526 SUB TOLER,RTEMP1 ;YES-OK-CHECK LOWEST READING.
6532 034344 163737 034512 034526 SUB TOLER,RTEMP1
6533 034352 023737 034520 034526 CMP RLOW,RTEMP1
6534 034360 002415 BLT ERAV1 ;NO-REPORT ERROR.
6535 034362 005737 034512 TST TOLER ;DOES OPERATOR WISH "FORCED" TYPEOUT?
6536 034366 001412 BEQ ERAV2 ;IF SO-DO IT
6537 034370 062737 000002 034464 AVERRN: ADD #2,CHAN1 ;SET TO DO NEXT CHAN-BUT
6538 034376 005237 034466 INC CHAN ;IF DOEN ALL CHANS-EXIT.
6539 034402 023737 034466 034476 CMP CHAN,CHANF ;OTHERWISE LOOP.
6540 034410 003675 BLE AVERRL
6541 034412 000207 RTS PC
6542
6543 ;*ERROR REPORTER
6544
6545 034414 ERAV1:
6546 034414 ERAV2:
6547 034414 005737 034540 TST REPOR ;REPORT ERROR?
6548 034420 001002 BNE 1$
6549 034422 000137 034370 JMP AVERRN
6550 034426 1$:
6551 ;GO TYPE--OCTAL ASCII(AL)
6552 034426 042737 170000 034520 BIC #170000,RLOW ;GO TYPE--OCTAL ASCII(AL)
6553 034434 042737 170000 034516 BIC #170000,AVTKN ;GO TYPE--OCTAL ASCII(AL)
6554 034442 042737 170000 034522 BIC #170000,RHIGH ;GO TYPE--OCTAL ASCII(AL)
6555
6556 034450 104020 ERROR 20
6557 034452 000746 BR AVERRN
6558
6559 ;*
6560 ;*POINTERS USED BY REPEATIBILITY TEST
6561 ;*
6562
6563
6564 034454 000000 CHANU: 0
6565 034456 000000 CHAN7:0
6566 034460 000000 NA07:0
6567 034462 000000 NA17:0
6568 034464 000000 CHAN1: 0 ;LEFT JUSTIFIED CURRENT CHANNELL.
6569 034466 000000 CHAN: 00 ;CURRENT CHANNELL
6570 034470 000000 GAIN: 00 ;CURRENT GAIN
6571 034472 000000 ADWD: 00 ;WORD SENT TO A/D
6572 034474 000000 CHANS: 00 ;STARTING CHANNELL
6573 034476 000000 CHANF: 00 ;LAST CHANNELL
6574 034500 000000 CHANSR: 0
6575 034502 000000 CHANFR: 0
6576 034504 000000 CHANNO: 0
6577 034506 000000 SAMCNT: 0 ;SAMPLE COUNT

```

6578	034510	000000		SAMOFF:	0			; SAMPLE OFFSET
6579	034512	000000		TOLER:	0			; TOLERANCE BEFORE ERROR IS REPORT
6580	034514	000000		AVEXP:	0			; EXPECTED AVERAGE
6581	034516	000000		AVTKN:	0			; AVERAGE OF SAMPLES TAKEN
6582	034520	000000		RLOW:	0			; LOWEST SAMPLE TAKEN
6583	034522	000000		RHIGH:	0			; HIGHEST SAMPLE TAKEN
6584	034524	000000		RTEMP:	0			
6585	034526	000000		RTEMP1:	0			
6586	034530	000000		REPMAN:	0			; SECTION RESERVED FOR
6587	034532	000000			0			; MANUAL REPEATIBILITY
6588	034534	000000			0			; WHERE OPERATOR ENTERS
6589	034536	000000			0			; IN INFORMATION
6590	034540	000000		REPOR:	0			
6591	034542	000000	000		0			
6592	034544	000000			0			
6593								
6594	034546	04401	043702	MAKEI:	TYPE, M0			; WELCOME MESSAGE
6595	034552	104411			RDLIN			
6596	034554	005037	001200		CLR	\$TESTN		
6597	034560	012600			MOV	(SP)+, R0		
6598	034562	121027	000117		CMPB	(R0), #'0		
6599	034566	001002			BNE	1\$		
6600	034570	000137	035516		JMP	LOOPJ		
6601	034574	104401	042615	1\$:	TYPE, M1			
6602	034600	104411			RDLIN			
6603	034602	012600			MOV	(SP)+, R0		
6604	034604	111037	001772		MOVB	(0), \$VERSN		
6605	034610	121027	000115		CMPB	(0), #'M		
6606	034614	001403			BEQ	2\$		
6607	034616	121027	000104		CMPB	(0), #'D		
6608	034622	001364			BNE	1\$		
6609	034624	104401	043030	2\$:	TYPE, M2			
6610	034630	104412			RDOCT			
6611	034632	012637	044114		MOV	(SP)+, KWT+2		
6612	034636	104401	043051		TYPE, M3			
6613	034642	104412			RDOCT			
6614	034644	012637	044116		MOV	(SP)+, KWT+4		
6615	034650	012737	000001	001124	3\$:	MOV	#1, \$GDDAT	
6616	034656	122737	000104	001772		CMPB	#'D, \$VERSN	
6617	034664	001403			BEQ	4\$		
6618	034666	104401	043114		TYPE, M4			
6619	034672	104412			RDOCT			
6620	034674	012637	001124		4\$:	MOV	(SP)+, \$GDDAT	
6621	034700	123727	001124	000011		CMPB	\$GDDAT, #11	
6622	034706	103360			BHIS	3\$		
6623	034710	013737	001124	001754		MOV	\$GDDAT, MYTEMP	
6624	034716	012737	000001	001126		MOV	#1, \$BDDAT	
6625	034724	012701	055066			MOV	#JOB0, R1	
6626	034730	012702	066000			MOV	#66000, R2	; FREE CORE .-
6627	034734	010237	056246			MOV	R2, FRECOR	
6628	034740	013702	056246			MOV	FRECOR, R2	
6629	034744	104401	043134		LIPED:	TYPE, M5		
6630	034750	013746	001126			MOV	\$BDDAT, -(SP)	:: SAVE \$BDDAT FOR TYPEOUT
6631	034754	104402				TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)

6632	034756	104401	043144	TYPE, M6	
6633	034762	104401	043154	TYPE, M7	
6634	034766	104412		RDOC↑	
6635	034770	012611		MOV (SP)+, (1)	; STORE MODE WORD
6636	034772	104401	043212	TYPE M8	
6637	034776	104412		RDOC↑	
6638	035000	012661	000002	MOV (SP)+, 2(1)	
6639	035004	104401	043347	TYPE, M10	; # OF BUFFERS
6640	035010	104412		RDOC↑	
6641	035012	112661	000007	MOVB (SP)+, 7(1)	
6642	035016	001772		BEQ 1\$	
6643	035020	012737	035664 000004	MOV #ERTOUT, 2#4	; IN CASE OF BAD BUFFER AREA
6644	035026	116137	000007 001202	MOVB 7(1), \$PASS	
6645	035034	105361	000007	DECB 7(1)	
6646	035040	010103		MOV R1, R3	
6647	035042	016104	000002	MOV 2(1), R4	
6648	035046	006304		ASL R4	
6649	035050	010261	000010	MOV R2, 10(1)	
6650	035054	104401	035062	TYPE 65\$:: TYPE ASCIZ STRING
6651	035060	000410		BR 64\$:: GET OVER THE ASCIZ
6652				64\$: .ASCIZ <200>#BUFFER ADDR. #	
6653	035102				
6654	035102	010246		MOV R2, -(SP)	:: SAVE R2 FOR TYPEOUT
6655	035104	104402		TYPOC	:: GO TYPE--OCTAL ASCII(ALL DIGITS)
6656	035106	060402		ADD R4, R2	
6657	035110	005712		TST (2)	
6658	035112	005337	001202	DEC \$PASS	
6659	035116	001403		BEQ 3\$	
6660	035120	062701	000004	ADD #4, R1	
6661	035124	000751		BR 2\$	
6662	035126	010301		MOV R3, R1	
6663	035130	104401	043244	TYPE, M9	; USER SW=
6664	035134	016346	000004	MOV 4(R3), -(SP)	:: SAVE 4(R3) FOR TYPEOUT
6665	035140	104402		TYPOC	:: GO TYPE--OCTAL ASCII(ALL DIGITS)
6666	035142	104401	043426	TYPE, M12	; BUFFER OVERRUN FATAL?
6667	035146	104411		RDLIN	
6668	035150	012600		MOV (SP)+, R0	
6669	035152	142710	000240	BICB #240, (0)	
6670	035156	122710	000131	CMPB #'Y, (0)	
6671	035162	001004		BNE 6\$	
6672	035164	042761	100000 000006	BIC #BIT15, 6(1)	
6673	035172	000406		BR 7\$	
6674	035174	121027	000116	CMPB (0), #'N	
6675	035200	001360		BNE 5\$	
6676	035202	052761	100000 000006	BIS #BIT15, 6(1)	
6677	035210	005071	000004	CLR 24(1)	; CLEAR USER WORD.
6678	035214	104401	043551	TYPE, M14	; DELAY BEFORE START
6679	035220	104412		RDOC↑	
6680	035222	012661	000054	MOV (SP)+, 54(1)	
6681	035226	104401	043757	TYPE, M20	; CHAN ADDR WORD
6682	035232	104412		RDOC↑	
6683	035234	012661	000056	MOV (SP)+, 56(1)	
6684	035240	104401	043575	TYPE, M15	; # OF CHAN
6685	035244	104412		RDOC↑	

6686	035246	012661	000060		MOV	(SP)+,60(1)	
6687	035252	104401	044003		TYPE	M21	;CLOCK RATE
6688	035256	104412			RDOCT		
6689	035263	012661	000062		MOV	(SP)+,62(1)	
6690	035264	032711	140000		BIT	#BIT14!BIT15,(1)	;EVENT OR START MASK
6691	035270	001453			BEQ	9\$	
6692	035272	104401	044020		TYPE	M22	
6693	035276	104412			RDOCT		
6694	035300	012661	000064		MOV	(SP)+,64(1)	
6695	035304	104401	044051		TYPE	M23	
6696	035310	104412			RDOCT		
6697	035312	012661	000066		MOV	(SP)+,66(1)	
6698	035316	104401	044067		TYPE	M24	
6699	035322	104412			RDOCT		
6700	035324	012661	000070		MOV	(SP)+,70(1)	
6701	035330	032711	001400		BIT	#BIT8!BIT9,(1)	;RANDOM CHANNEL?
6702	035334	001031			BNE	9\$;NO EXIT THIS LOOP
6703	035336	016037	000060	001202	MOV	60(0), \$PASS	;YES GET NUMBER OF CHANS.
6704	035344	012703	000062		MOV	#(JOB1-JOB0-34),R3	
6705	035350	060103			ADD	R1,R3	
6706	035352	010361	000050		MOV	R3,50(1)	
6707							
6708	035356			13\$:	TYPE	,67\$::TYPE ASCIZ STRING
6709	035356	104401	035364		BR	66\$::GET OVER THE ASCIZ
6710	035362	000407					
6711				67\$:	.ASCIZ	<200>#RANDOM CH.	=#
6712	035402			68\$:			
6713	035402	104412			RDOCT		
6714	035404	012623			MOV	(SP)+,(3)+	
6715	035406	005337	001202		DEC	\$PASS	
6716	035412	001361			BNE	13\$	
6717	035414	012713	140000		MOV	#BIT15!BIT14,(3)	;ADD EOC MARKER.
6718							
6719	035420			9\$:			
6720							
6721	035420	104401	043275		TYPE	M97	
6722	035424	104412			RDOCT		
6723	035426	012661	000074		MOV	(SP)+,74(1)	
6724	035432	032711	000200		BIT	#BIT7,(1)	;OUTPUT DEVICE?
6725	035436	001404			BEQ	10\$;NO-CONTINUE
6726	035440	104401	043660		TYPE	M18	;MAKE BUFFER
6727	035444	104401	043724		TYPE	M19	;PRESS CONTINUE
6728	035450	010237	056246		MOV	R2,FRECOR	
6729	035454	023727	056246	000000G	CMP	FRECOR,#DRLPX0	;CORE EXCEEDED?
6730	035462	103100			BHIS	ERTOUT	;YES-ERROR MESSAGE!
6731	035464	123737	001126	001124	CMPB	\$BDDAT,\$GDDAT	;DONE ALL?
6732	035472	001411			BEQ	11\$	
6733	035474	005237	001126		INC	\$BDDAT	
6734	035500	012703	055204		MOV	#JOB1,R3	;GET JOB SIZ
6735	035504	162703	055066		SUB	#JOB0,R3	
6736	035510	060301			ADD	R3,R1	;UPDATE JOB POINTER
6737	035512	000137	034744		JMP	L1PED	;LOOP FOR NEXT JOBS DATA
6738							
6739	035516			11\$:			

```

6740 035516 004737 036026 LOOPJ: JSR PC,FLASH
6741 6742 ;;REPORT DATA
6743 035522 104401 035530 TYPE 65$ ;;TYPE ASCIZ STRING
6744 035526 000414 BR 64$ ;;GET OVER THE ASCIZ
6745 ;;65$: .ASCIZ <200>#REPORT DATA FOR JOB 0?#
6746 64$:
6747 035560 RDLIN
6748 035562 104411 MOV (SP)+,R0
6749 035564 121027 000131 CMPB (0),#Y
6750 035570 001402 BEQ 2$
6751 035572 000137 034546 1$: JMP MAKEI
6752
6753 035576 012701 055066 2$: MOV #JOB0,R1
6754 035602 013700 001754 MOV MYTEMP,R0
6755 035606 016137 000060 034454 3$: MOV 60(1),CHANU ;GET NU. OF CHANS.
6756 035614 005003 CLR R3
6757
6758 035616 013702 034454 4$: MOV CHANU,R2 ;NU OF CHANS
6759 035622 010337 034474 MOV R3,CHANS
6760 035626 010337 034476 MOV R3,CHANF
6761
6762 035632 005237 034540 INC REPOR
6763 035636 004737 034146 JSR PC,AVERR
6764 035642 005203 INC R3
6765 035644 005302 DEC R2
6766 035646 001365 BNE 4$
6767 035650 062701 000116 ADD #<JOB1-JOB0>,R1
6768 035654 005300 DEC R0
6769 035656 001353 BNE 3$
6770 035660 000744 BR 1$
6771 035662 000000 HALT
6772
6773
6774 035664 ERTOUT:
6775 035664 104401 035672 TYPE 65$ ;;TYPE ASCIZ STRING
6776 035670 000434 BR 64$ ;;GET OVER THE ASCIZ
6777 ;;65$: .ASCIZ <200>#INSUFFICIENT CORE TO DO REQUEST,PLEASE SPECIFY SMALLER#
6778 64$:
6779 035762 TYPE 67$ ;;TYPE ASCIZ STRING
6780 035762 104401 035770 BR 66$ ;;GET OVER THE ASCIZ
6781 035766 000415 ;;67$: .ASCIZ <200>#BUFFER OR LESS BUFFERS#
6782 66$:
6783 036022 JMP MAKEI
6784 036022 000137 034546
6785
6786 ;*
6787 ;*FLASH! ROUTINE TO EXERCISE 1-8 JOBS UNTIL DONE.
6788 ;* REQUIRED:
6789 ;* $BERSN MUST CONTAIN ASCII DOR M FOR DEDICATED OR MULTIUSER.
6790 ;* KWT PRE SET UP TO CLOCK STATS
6791 ;* JOB0-JOB7 PRE SET UP
6792 ;* $GDDAT= NO. OF JOBS TO EXERCISE
6793 ;*
6794 ;* CALL= JSR PC, FLASH

```


; * RETURNS WHEN ALL DONE.

```

6794
6795
6796 036026 FLASH:
6797
6798 036026 004737 040626 2$: JSR PC,SRESET
6799 036032 012777 054130 143436 MOV #DEVST,ALPADL ;SET ADDR. OF DEVICE LIST IN RDA WORD
6800 036040 005037 054130 CLR DEVST ;INDICATE SIZING
6801 036044 113737 001512 054131 MOV#B VERSN,DEVIST+1 ;SET VERSION NUMBER
6802
6803 036052 152777 000001 143406 BISB #1,ALPCI ;SET GO.
6804 036060 004737 040542 JSR PC,SDELAY
6805 036064 012777 044112 143404 MOV #KWT,ALPADL ;START CLOCK
6806 036072 052777 000001 143366 BIS #BIT0,ALPCI
6807 036100 012700 054520 MOV #TODOQ,RO
6808
6809 036104 005020 3$: CLR (0)+ ;CLEAR QUE AREA.
6810 036106 020027 055004 CMP RO,#DONEC
6811 036112 001374 BNE 3$
6812 036114 013700 001124 MOV $GDDAT,RO ;GET NO. OF JOBS
6813 036120 012701 054520 MOV #TODOQ,R1 ;GET QUES AREA
6814 036124 012702 055066 MOV #JOB0,R2 ;PICK UP 1ST JOB ADDRESS
6815 036130 012703 055204 MOV #JOB1,R3 ;CALCULATE ADDR DIF.
6816 036134 162703 055066 SUB #JOB0,R3
6817 036140 010037 054640 MOV RO,TODOC ;REMEMBER HOW MANY JOBS ARE QUEUED
6818 036144 012700 000010 MOV #8,RO
6819 036150 005712 4$: TST (2) ;ANYTHING IN THIS JOB??
6820 036152 001401 BEQ 6$
6821 036154 010221 6$: MOV R2,(1)+ ;STORE FIRST JOB
6822
6823 036156 060302 ADD R3,R2 ;POINT TO NEXT JOB
6824 036160 005300 DEC RO ;DONE ALL JOBS?
6825 036162 001372 BNE 4$ ;NO-LOOP.
6826 036164 005037 001126 CLR $BDDAT
6827 036170 063737 002014 001126 ADD FUDGE,$BDDAT ;DON'T WAIT FOR CLOCK JOBS(DR11K ONLY)
6828 036176 005037 002014 CLR FUDGE
6829 036202 013700 001556 MOV VECT1,RO
6830 036206 012720 03641C MOV #OUPSRV,(0)+
6831 036212 012720 000340 MOV #340,(0)+
6832 036216 012720 036334 MOV #INPSRV,(0)+
6833 036222 012720 000340 MOV #340,(0)+
6834 036226 052777 000100 143232 BIS #BIT6,ALPCI
6835 036234 052777 000100 143230 BIS #BIT6,ALPCO
6836 036242 013737 001124 001774 MOV $GDDAT,TAXING ;GET NO OF REQUESTS.
6837 036250 062737 000200 001774 ADD #200,TAXING ;MUL X2 TO GET TIME.
6838 036256 006337 001774 ASL TAXING
6839 036262 005037 177776 CLR PS
6840 036266 023737 001124 001126 5$: CMP $GDDAT,$BDDAT ;DONE ALL JOBS?
6841 036274 001414 BEQ 7$
6842 036276 004737 040542 JSR PC,SDELAY
6843 036302 005337 001774 DEC TAXING
6844 036306 001367 BNE 5$
6845 036310 017737 143152 001124 MOV ALPCI,$GDDAT
6846 036316 017737 143150 001126 MOV ALPCO,$BDDAT
6847 036324 104017 ERROR 17 ;JOB(S) FAILED TO FINISH (LPA FAULT)

```

```

6848 036326          7$:
6849 036326 004737 040626      JSR      PC,SRESET
6850 036332 000207          RTS      PC
6851 036334 005737 054640      INPSRV: TST      TODOC          ;ANYTHING TO DO?
6852 036340 001422          BEQ      IEX
6853 036342 010046          MOV      RO,-(SP)
6854 036344 012700 054520      MOV      #TODOC,RO
6855 036350 005720          1$:      TST      (0)+
6856 036352 001776          BEQ      1$
6857 036354 014077 143116      MOV      -(0),ALPADL          ;SAVE RDA ADDR.
6858 036360 005010          CLR      (0)                  ;ZERO THIS JOB
6859 036362 005337 054640      DEC      TODOC
6860 036366 001003          BNE      2$
6861 036370 042777 000100 143070 BIC      #BIT6,ALPCI          ;DON'T ALLOW ANOTHER INTERRUPT.
6862 036376          2$:
6863 036376 052777 000001 143062 BIS      #BIT0,ALPCI          ;SET GO!
6864 036404 012600          MOV      (SP)+,RO
6865 036406 000002          IEX:   RTI
6866
6867 036410 010046          OUPSRV: MOV      RO,-(SP)          ;SAVE REGS.
6868 036412 010146          MOV      R1,-(SP)
6869 036414 010246          MOV      R2,-(SP)
6870 036416 010346          MOV      R3,-(SP)
6871 036420 010446          MOV      R4,-(SP)
6872 036422 005237 001774          INC      TAXING
6873
6874 036426 017737 143044 001770      MOV      ALPADL,ALPADL          ;SAVE RDA ADDR.
6875 036434 017737 143032 001764      MOV      ALPC0,ALPC0          ;GET CONTROL IN REG.
6876 036442 013700 001764          MOV      ALPC0,RO            ;NOW LETS GET USER NUMBER.
6877 036446 042700 177770          BIC      #C<7>,RO            ;RID OF REST OF JUNK.
6878 036452 010037 001756          MOV      RO,USJNO           ;GET USJNO
6879 036456 010003          MOV      RO,R3
6880 036460 006300          ASL      RO
6881 036462 013701 001770          MOV      ALPADL,R1
6882 036466 105737 001765          TSTB    ALPC0+1              ;SEE IF BIT15 SET SHOWING ERROR
6883 036472 100024          BPL      NER                  ;IF NO ERROR,PROCEED.
6884
6885 036474 017737 142766 001762      MOV      ALPCI,ALPCI          ;ELSE REPORT AN ERROR.
6886 036502 113737 001765 001766      MOVE    ALPC0+1,ALPC0        ;FIX STATUS REG SO ITS EASIER TO READ.
6887 036510 105037 001767          CLRB    ALPC0+1              ;ONLY LOW BYTE.
6888 036514 122737 000250 001766      CMPB    #250,ALPC0           ;LEGAL ERROR. BIT 14 SET IN USW
6889 036522 001003          BNE      40$
6890 036524 005237 001126          INC      $BDDAT
6891 036530 000440          BR      OUPEX
6892
6893 036532 104021          40$:   ERROR 21                ;LPA-11 MICRO CODE REPORTED AN ERROR
6894
6895 036534 013737 001124 001126      MOV      $GDDAT,$BDDAT        ;FIX COUNT FOR EXIT TO MAIN LINE FLASH.
6896 036542 000433          BR      OUPEX
6897
6898 036544 001005          NER:   BNE      10$
6899 036546 005071 000004          CLR      24(1)                ;CLEAR USW
6900 036552 013760 001770 054642      MOV      ALPADL,TABLE(0)      ;SAVE RDA ADDR. FOR FUTURE REF.
6901

```

```

6902 036560          10$:
6903 036560 016001 054642      MOV     TABLE(0),R1      ;GET RDA ADDR FROM TABLE
6904 036564 016103 000004      MOV     4(1),R3          ;GET ADDR OF USW
6905 036570 116302 000001      MOVB   1(3),R2          ;GET LAST BUFFER USED
6906 036574 005202              INC     R2              ;UPDATE FOR NEXT BUFFER
6907 036576 042702 177770      BIC    #177770,R2       ;CLEAN OFF JUNK
6908 036602 116104 000007      MOVB   VBM(1),R4        ;GET LAST BUFFER MASK.
6909 036606 042704 177770      BIC    #177770,R4        ;CLEAN OFF JUNK
6910
6911 036612 042702 177770      30$:  BIC    #1C<7>,R2      ;VALID OFFSET.
6912 036616 110263 000001      MOVB   R2,1(3)
6913 036622 120204              CMPB   R2,R4            ;LAST BUFFER ?
6914 036624 003402              BLE    OUPEX            ;NO-BRANCH
6915 036626 052713 040000      BIS    #40000,(3)      ;SET STOP JOB BIT
6916 036632 042777 000200 142632 OUPEX: BIC    #BIT7,2LPC0      ;CLEAR READY OUT
6917
6918 036640 012604              MOV    (SP)+,R4        ;RESTORE REGISTERS
6919 036642 012603              MOV    (SP)+,R3
6920
6921 036644 012602              MOV    (SP)+,R2
6922 036646 012601              MOV    (SP)+,R1
6923 036650 012600              MOV    (SP)+,R0
6924 036652 000002              RTI
6925

```

```

6926
6927
6928
6929
6930
6931
6932
6933
6934
6935
6936
6937
6938
6939
6940
6941
6942
6943 036654
6944 036654 013746 000004
6945
6946 036660 000413
6947
6948
6949
6950
6951
6952
6953
6954
6955
6956
6957
6958
6959 036662 012737 036706 000004
6960 036670 005237 170000
6961 036674 104401 036702
6962 036700 000401
6963
6964 036704
6965 036704 000401
6966 036706 022626
6967 036710 012637 000004
6968 036714 005037 037532
6969 036720 004537 037534
6970 036724 000000G
6971
6972 036726 052777 040000 142532
6973
6974 036734
6975
6976 036734 010146
6977 036736 005001
6978 036740 005201
6979 036742 001376

```

```

; *
; * THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
; * FIRST WE WILL LOAD MICROCODE INTO KMC-11
; * NEXT WE WILL INIT BOTH UPROCESSORS
; * THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
; * THE ORDER OF LOAD IS DETERMINED BY THE USER.
; *
; *      CALL=   JSR      R5,$LPAI
; *              .WORD   0              ; ADDR. OF DEVICE ADDRESS.
; * ROUTINES REQUIRED:  .LOADLP
; * PROGRAMS REQUIRED:  DRLPX2
; *
; *
; *              ; RETURNS WITH $AERR=1 IF SLAVE
; *              ; MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
; *
$LPAI:
MOV      4,-(SP)
BR       31$
; FIELD DOES NOT HAVE A BUS SWITCH TO
; WORRY ABOUT, SO WE WILL UNCONDITIONALLY
; BRANCH AROUND THE NEXT CODE THAT
; WORKS BASED ON A BUS SWITCH.
; CODE LEFT IN HERE FOR IN HOUSE
; PERSONAL WHO MAY PATCH THIS BRANCH
; INSTRUCTION TO A <NOP> OCTAL <240>
; IN ORDER TO RUN PROGRAM WITH A SWITCH.
; NOTE THIS "SWITCH" IS A PIECE OF INHOUSE
; TEST EQUIPMENT ONLY IT CONNECTS
; THE UNIBUS TO THE I/O BUS FOR
; CERTAIN TESTING.
MOV      #30$ 4
INC      170000
TYPE     65$
BR       64$
; ; TYPE ASCIZ STRING
; ; GET OVER THE ASCIZ
; ; 65$: .ASCIZ <7>##
; ; 64$:
BR       31$
30$:    CMP      (SP)+,(SP)+
31$:    MOV      (SP)+,4
; ALL THIS JUNK MUST BE REMOVED!!
CLR      $AERR
JSR      R5,$LOAD
; LOAD MICRO-CODE.
; FILE "DRLPX2.OBJ"
; .WORD   DRLPX2
BIS      #BIT14,$KMA00
; ISSUE KMC+DMC INIT.
1$:
; "HANGS" HERE THEN KMC-11 ERROR.
MOV      R1,(SP)
CLR      R1
2$:    INC      R1
; STALL FOR DMC-UP
BNE     2$

```

```

6980 036744 012777 104000 142514      MOV      #BIT15!BIT11, @KMA00      ;SET RUN, AND ENABLE ARBITRATION.
6981 036752 105201      INCB    R1
6982 036754 001376      BNE     25$
6983
6984 036756 032777 000040 142502      BIT      #BITS, @KMA00      ;SLAVE READY? (READING IPBM SR)
6985 036764 001401      BEQ     3$
6986
6987 036766 104000      ERROR
6988
6989 036770 012777 000004 142474      MOV      #4, @KMA02      ;READ FAST PATH
6990 036776 004537 040444      JSR     RS, $TOUT      4$: ;-TOUT-CHECK FOR TIMEOUT
6991
6992 037002 104000      ERROR
6993
6994
6995
6996
6997
6998
6999 037004 000774      BR      4$
7000
7001
7002 037006 122777 000377 142456      CMPB    #377, @KMA02      ;/RETURNS HERE-FROM-TIMED OUT.
7003 037014 001370      BNE     4$              ;WAIT TILL KMC DONE COMMAND.
7004 037016 122777 000377 142452      CMPB    #377, @KMA04      ;IF FAST PATH=377 THEN ERROR.
7005 037024 001001      BNE     35$
7006 037026 104000      ERROR
7007
7008
7009 037030 122777 000004 142440      CMPB    #4, @KMA04      ;IS THIS THE CORRECT VERSION OF MICRO-CODE?
7010 037036 001543      BEQ     5$              ;YES-CONTINUE.
7011 037040 005227 177777      INC     #-1
7012 037044 001140      BNE     5$
7013 037046 005227 177777      INC     #-1
7014 037052 001135      BNE     5$
7015 037054 104401 037062      TYPE    ,67$           ;;TYPE ASCIZ STRING
7016 037060 000440      BR      66$           ;;GET OVER THE ASCIZ
7017
7018 037162      ;;67$: .ASCIZ <200>"W A R N I N G THIS PROGRAM WAS DESIGNED TO RUN WITH VERSION 4"
7019 037162 104401 037170      66$: TYPE    ,69$           ;;TYPE ASCIZ STRING
7020 037166 000430      BR      68$           ;;GET OVER THE ASCIZ
7021
7022 037250      ;;69$: .ASCIZ <200>"MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED."
7023 037250 104401 037256      68$: TYPE    ,71$           ;;TYPE ASCIZ STRING
7024 037254 000434      BR      70$           ;;GET OVER THE ASCIZ
7025
7026 037346      ;;71$: .ASCIZ <200>"THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED."<200><200>
7027      70$:
7028 037346 112737 177777 037500      5$:  MOVB    #0-1, 11$      ;DAC CODE FOR SLAVE.
7029 037354 012501      MOV     (5)+, P1      ;GET NEXT DEVICE ADDR.
7030 037356 021:27 000000      6$:  CMP     (R1), #0      ;TERM REACHED?
7031 037362 001444      BEQ     10$
7032 037364 105237 037500      INCB    11$
7033 037370 113777 037500 142100      MOVB    11$, @KMA04      ;FIFO DATA

```

```

7034 037376 004737 037502      JSR    PC,20$      ;ISSUE SEND
7035 037402 112177 142070      MOVB   (R1)+,2KMAD4 ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
7036 037406 004737 037502      JSR    PC,20$      ;ISSUE SEND
7037 037412 112177 142060      MOVB   (R1)+,2KMAD4 ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
7038 037416 004737 037502      JSR    PC,20$
7039
7040 037422 032777 000002 142036 7$:  BIT    #BIT1,2KMAD0 ;WAIT FOR FIFO DATA
7041 037430 001374          BNE    7$          ;=1 NO DATA. =0 DATA.
7042 037432 112777 000002 142032      MOVB   #2,2KMAD2   ;READ FIFO.
7043
7044 037440          8$:
7045 037440 004537 040444      JSR    R5, $TOUT   ; -TOUT-CHECK FOR TIMEOUT
7046
7047 037444 104000          ERROR           ; /TIME-OUT ERROR
7048          ; /WE FAILED TO COMPLETE
7049          ; /CURRENT OPERATION.
7050          ; /CONTINUES IN THIS LOOP
7051          ; /WOULD MAKE US "HANG" HERE
7052
7053 037446 000774          BR      8$
7054
7055          ; /RETURNS HERE-FROM-TIMED OUT.
7056 037450 122777 000377 142014      CMPB   #377,2KMAD2 ;WAIT FOR READ.
7057 037456 001370          BNE    8$
7058 037460 105777 142012      TSTB   2KMAD4      ; WAS A ZERO RETURNED?
7059 037464 001734          BEQ    6$          ; YES GET NEXT ADDR.
7060          ; SLAVE WILL RETURN CODE 0 IF
7061 037466 005237 037532      INC    $AERR       ; DEV PRESENT. ELSE
7062          ; EXIT $AERR=1 IF SLAVE GIVES ERROR.
7063 037472 005041          CLR    -(1)        ; GET RID OF REFERENCE TO BAD ADDR.
7064 037474 012601          10$: MOV    (SP)+,R1
7065 037476 000205          RTS    R5          ; RETURN ALL ADDR. CHECKED.
7066
7067 037500 000000          11$: .WORD 0      ; HOLDS DAC CODE PLUS OFFSET
7068          ; TO SLAVES ADDR. TABLE.
7069
7070 037502 112777 000003 141762 20$: MOVB   #3,2KMAD2   ; ISSUE FIFO WRITE
7071 037510          21$:
7072 037510 004537 040444      JSR    R5, $TOUT   ; -TOUT-CHECK FOR TIMEOUT
7073
7074 037514 104000          ERROR           ; /TIME-OUT ERROR
7075          ; /WE FAILED TO COMPLETE
7076          ; /CURRENT OPERATION.
7077          ; /CONTINUES IN THIS LOOP
7078          ; /WOULD MAKE US "HANG" HERE
7079
7080 037516 000774          BR      21$
7081
7082          ; /RETURNS HERE-FROM-TIMED OUT.
7083 037520 122777 000377 141744      CMPB   #377,2KMAD2 ;KMC CODE WILL RETURN A "377"
7084 037526 001370          BNE    21$        ;WHEN DONE COMMAND.
7085 037530 000207          RTS    PC
7086
7087 037532 000000          $AERR: .WORD 0   ; =0 IF ADDR. LIST OK, =1 IF BAD.

```

```

7088
7089
7090
7091
7092
7093
7094
7095
7096
7097 037534 010446          $LOAD: MOV R4, -(SP)      ;SAVE R4.
7098 037536 010046          MOV R0, -(SP)      ;SAVE R0.
7099 037540 012500          1$:  MOV (5)+, R0      ;GET PROG. ADDR.
7100 037542 005077 141720  CLR @KMAD0         ;CLEAR CSR
7101 037546 005077 141724  CLR @KMAD4         ;CLEAR CRAM ADDR.
7102 037552 052777 002000 141706 2$:  BIS #2000, @KMAD0  ;SELECT CRAM.
7103 037560 012077 141716  MOV (0)+, @KMAD6   ;WRITE DATA.
7104 037564 052777 020000 141674  BIS #20000, @KMAD0 ;SET CRAM WRITE
7105 037572 005077 141670  CLR @KMAD0         ;DISABLE CRAM.
7106 037576 005277 141674  INC @KMAD4         ;UPDATE CRAM ADDR.
7107 037602 021027 177777  CMP (0), #-1      ;ALL DONE?
7108 037606 001361  BNE 2$              ;NO LOOP.
7109 037610 005077 141662  CLR @KMAD4         ;CLEAR CRAM ADDR.
7110 037614 016500 177776  MOV -2(5), R0     ;GET MICRO CODE ADDR.
7111
7112 037620 052777 002000 141640 3$:  BIS #2000, @KMAD0  ;SELECT CRAM
7113 037626 022077 141650  CMP (R0)+, @KMAD6 ;DATA OK?
7114 037632 001013  BNE 5$              ;NO - REPORT AN ERROR.
7115 037634 021027 177777  CMP (0), #-1      ;ALL DONE?
7116 037640 001405  BEQ 4$              ;YES - EXIT
7117 037642 005077 141620  CLR @KMAD0         ;NO - DESELECT CRAM.
7118 037646 005277 141624  INC @KMAD4         ;UPDATE CRAM ADDR.
7119 037652 000762  BR 3$
7120
7121 037654 012600          4$:  MOV (SP)+, R0      ;RESTORE R0
7122 037656 012604          MOV (SP)+, R4      ;RESTORE R4
7123 037660 000205          RTS R5             ;EXIT
7124
7125 037662          5$:  ;COME HERE ON LOAD ERROR
7126 037662 005745          TST -(5)
7127 037664 105204          INCB R4           ;UPDATE ERROR COUNTER.
7128 037666 100324          BPL 1$           ;IF NOT TOO MANY, TRY AGAIN.
7129 037670 000000          HALT           ;MICRO CODE LOAD ERROR.
7130
7131 037672 000722          BR 1$           ;KMC-11 FAULT. YOU COULD TRY
7132
7133
7134
7135
7136
7137
7138
7139
7140
7141
; *THIS ROUTINE ISSUES A WRITE COMMAND TO THE LPA-11
; *
; * CALL = JSR R5, $TLKW

```

```

7142                ;*                .WORD 0                ;OFFSET OF DEVICE ADDR.
7143                ;*                .WORD 0                ;DATA TO BE WRITTEN
7144                ;*
7145 037674 010046          $TLKW: MOV      RO, -(SP)          ;SAVE RO
7146 037676 012500          MOV      (5)+, RO          ;GET DEVICE OFFSET
7147 037700 052700 000340  BIS      #340, RO          ;ADD WRITE CODE.
7148 037704 004737 040156  JSR      PC, $LPW          ;WAIT FOR FAST PATH READY
7149 037710 010037 040002  MOV      RO, W1
7150 037714 010077 141556  MOV      RO, @KMA04
7151 037720 112777 000005 141544  MOVB   #5, @KMA02          ;ISSUE FAST PATH WRITE
7152 037726 004737 040156  JSR      PC, $LPW          ;WAIT FOR RDY
7153 037732 011537 040004  MOV      (5), W2
7154 037736 112577 141534  MOVB   (5)+, @KMA04      ;WRITE LOW BYTE DATA.
7155
7156 037742 112777 000005 141522  MOVB   #5, @KMA02          ;FP WRITE
7157 037750 004737 040156  JSR      PC, $LPW
7158 037754 111537 040006  MOVB   (5), W3
7159 037760 112577 141512  MOVB   (5)+, @KMA04      ;WRITE HIGH BYTE
7160 037764 112777 000005 141500  MOVB   #5, @KMA02
7161 037772 004737 040156  JSR      PC, $LPW
7162 037776 012600          MOV      (SP)+, RO
7163 040000 000205          RTS      R5                ;EXIT DONE.
7164 040002 000000          W1:    0
7165 040004 000000          W2:    0
7166 040006 000000          W3:    0
7167
7168                ;*
7169                ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
7170                ;*
7171                ;*      CALL = JSR      R5, $TLKr
7172                ;*                .WORD 0                ;OFFSET OF DEVICE
7173                ;*                ;RETURNS HERE
7174                ;*DATA IN WORD $DATR
7175                ;*
7176
7177 040010 010046          $TLKR: MOV      RO, -(SP)          ;SAVE RO
7178 040012 012500          MOV      (5)+, RO          ;GET OFFSET
7179 040014 052700 000300  BIS      #300, RO          ;ADD READ CODE
7180 040020 004737 040156  JSR      PC, $LPW          ;WAIT TILL READY
7181 040024 110077 141446  MOVB   RO, @KMA04
7182 040030 112777 000005 141434  MOVB   #5, @KMA02          ;ISSUE WRITE FP
7183 040036 004737 040156  JSR      PC, $LPW
7184 040042 010037 040152  MOV      RO, RD1
7185 040046          1$:    JSR      R5, $TOUT          ; -TOUT-CHECK FOR TIMEOUT
7186 040046 004537 040444
7187
7188 040052 104000          ERROR          ;/TIME-OUT ERROR
7189                ;/WE FAILED TO COMPLETE
7190                ;/CURRENT OPERATION.
7191                ;/CONTINUES IN THIS LOOP
7192                ;/WOULD MAKE US "HANG" HERE
7193
7194 040054 000774          BR      1$
7195

```



```

7196                                     ;/RETURNS HERE-FROM-TIMED OUT.
7197 040056 032777 000040 141402        BIT    #BITS,@KMADD    ;FAST PATH GOT DATA?
7198 040064 001370                                     BNE    1$
7199 040066 112777 000004 141376        MOVB   #4,@KMAD2     ;ISSUE FAST PATH READ
7200 040074 004737 040156                                     JSR    PC,$LPW
7201 040100 117737 141372 040154        MOVB   @KMAD4,$DATR  ;GET LOW BYTE
7202 040106                                     2$:
7203 040106 004537 040444        JSR    R5,$TOUT     ;-TOUT-CHECK FOR TIMEOUT
7204                                     ERROR
7205 040112 104000                                     ;/TIME-OUT ERROR
7206                                     ;/WE FAILED TO COMPLETE
7207                                     ;/CURRENT OPERATION.
7208                                     ;/CONTINUES IN THIS LOOP
7209                                     ;/WOULD MAKE US "HANG" HERE
7210
7211 040114 000774        BR          2$
7212
7213                                     ;/RETURNS HERE-FROM-TIMED OUT.
7214 040116 032777 000040 141342        BIT    #BITS,@KMADD    ;FAST PATH READY?
7215 040124 001370                                     BNE    2$
7216 040126 112777 000004 141336        MOVB   #4,@KMAD2     ;ISSUE FAST PATH READ
7217 040134 004737 040156                                     JSR    PC,$LPW
7218 040140 117737 141332 040155        MOVB   @KMAD4,$DATR+1 ;SAVE HIGH BYTE
7219 040146 012600        MOV     (SP)+,R0
7220 040150 000205        RTS     R5
7221 040152 000000        RD1: 0
7222 040154 000000        $DATR: .WORD 0
7223
7224                                     ; THIS ROUTINE WAITS FOR KMC-CODE TO BECOME READY AS WELL
7225                                     ; AS FAST PATH TO BE READ.
7226
7227                                     ; CALL = JSR PC,$LPW
7228
7229                                     ; IT WILL TIME OUT IF TOO MUCH TIME IS TAKEN BY
7230                                     ; THE MICRO-PROCESSORS AND REPORT AN ERROR, THEN HALT.
7231
7232
7233 040156 010146        $LPW: MOV     R1,-(SP)    ;SAVE R1
7234 040160 005001        CLR     R1
7235 040162 122777 000377 141302 1$:    CMPB   #377,@KMAD2   ;FINISHED INSTRUCTION?
7236 040170 001403        BEQ    2$
7237 040172 005201        INC     R1          ;TIME OUT?
7238 040174 001372        BNE    1$
7239 040176 000411        BR     10$
7240
7241 040200 032777 000020 141260 2$:    BIT    #BIT4,@KMADD  ;FAST PATH READ?
7242 040206 001403        BEQ    3$
7243 040210 005201        INC     R1          ;NO - TIME OUT?
7244 040212 001372        BNE    2$
7245 040214 000402        BR     10$        ;YES - REPORT AN ERROR
7246
7247 040216 012601        3$:  MOV     (SP)+,R1    ;RESTORE R1
7248 040220 000207        RTS     PC        ;EXIT
7249

```

```

7250 040222
7251 040222 104401 040230
7252 040226 000407
7253
7254 040246
7255
7256 040246 000000
7257 040250 000776
7258
7259
7260
7261
7262
7263
7264
7265
7266
7267
7268
7269
7270
7271
7272 040252 010046
7273 040254 010146
7274
7275 040256 012700 001514
7276 040262 005001
7277 040264 005710
7278 040266 001421
7279 040270 027520 000000
7280 040274 001402
7281 040276 005201
7282 040300 000771
7283
7284 040302 010137 040320
7285 040306 005725
7286 040310 013537 040322
7287 040314 004537 037674
7288 040320 000000
7289 040322 000000
7290 040324 012601
7291 040326 012600
7292 040330 000205
7293 040332 017520 000000
7294 040336 005010
7295 040340 004537 036654
7296 040344 001514
7297 040346 000755
7298
7299
7300
7301
7302
7303

```

```

10$: TYPE 65$ ;TYPE ASCIZ STRING
BR 64$ ;GET OVER THE ASCIZ
;65$: .ASCIZ <200>#LPA-11 FAULT#
64$:
11$: HALT ;LPA-11 FAULT RUN LPA-11
BR 11$ ;DIAGNOSTICS.

```

```

;*
;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE TO
;*A DEVICE ADDRESS ON THE I/O BUSS FOR WRITE ONLY.
;*
;* FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN USED
;* BEFORE. IF NOT WE HAVE TO INITIALIZE THE LPA WITH
;* THAT ADDRESS.
;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
;* $TLKw
;*
$OUTLP: MOV R0,-(SP) ;SAVE R0
MOV R1,-(SP) ;SAVE R1
MOV #.DVLS,R0 ;PROGRAM DEFINED LIST.
1$: CLR R1
TST (0) ;TERMINATOR REACHED?
BEQ 10$ ;YES NEXT STEP.
CMP @ (5),(0)+ ;MATCH WITH ADDR IN LIST?
BEQ 2$
INC R1
BR 1$
2$: MOV R1,3$ ;SAVE OFFSET, DEVICE KNOWN.
TST (5)+
MOV @ (5)+,4$ ;GET DATA TO BE WRITTEN
JSR RS,$TLKw ;DO WRITE
3$: .WORD 0 ;DEVICE OFFSET
4$: .WORD 0 ;DATA TO BE WRITTEN.
MOV (SP)+,R1
MOV (SP)+,R0
RTS RS
10$: MOV @ (5),(0)+ ;SAVE ADDR.
CLR (0)
JSR RS,$LPAI
.WORD .DVLS
BR 2$

```

```

;*
;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE
;*TO A DEVICE ADDR. ON THE I/O BUSS FOR READ ONLY.
;*
;*FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN

```

```

7304 ;*USED BEFORE. IF NOT, WE HAVE TO INITIALIZE THE LPA
7305 ;*WITH THE NEW ADDR.
7306 ;*WHEN THE ADDR IS KNOWN WE CAN DO OUTPUT THROUGH
7307 ;*$TLKR
7308 ;* CALL THROUGH MOVEI DATA,ADDR.
7309 ;* WHICH EQUALS:
7310 ;* JSR RS,$INLP
7311 ;* .WORD XX ADDR OF DEVICE
7312 ;* .WORD YY ADDR TO STORE READ DATA.
7313
7314 040350 010046 $INLP: MOV RO,-(SP) ;SAVE RO
7315 040352 010146 MOV R1,-(SP) ;SAVE R1
7316
7317 040354 012700 C01514 MOV #.DVLS,R0 ;PROG DEFINED ADDR. LIST.
7318 040360 005001 CLR R1
7319 040362 005710 1$: TST (0) ;EOL REACHED?
7320 040364 001420 BEQ 10$ ;YES - DEFINE NEW ADDR.
7321
7322 040366 027520 000000 CMP @ (5), (0)+ ;ADDR. MATCH?
7323 040372 001402 BEQ 2$
7324 040374 005201 INC R1
7325 040376 000771 BR 1$
7326
7327 040400 010137 040412 2$: MOV R1,3$ ;SAVE LIST OFFSET
7328 040404 005725 TST (5)+
7329 040406 004537 040010 JSR RS,$TLKR ;GO READ DEVICE
7330
7331 040412 000000 $OFS=. 3$: .WORD 0 ;OFFSET OF DEVICE
7332
7333 040414 013735 040154 MOV $DATR,@ (5)+ ;STORE DATA.
7334 040420 012601 MOV (SP)+,R1 ;RESTORE R1
7335 040422 012600 MOV (SP)+,R0 ;RESTORE R2
7336 040424 000205 RTS RS ;EXIT
7337
7338 040426 017520 000000 10$: MOV @ (5), (0)+
7339 040432 005010 CLR (0)
7340 040434 004537 036654 JSR RS,$LPAI
7341 040440 001514 .WORD .DVLS
7342 040442 000756 BR 2$
7343
7344 ;*$STOUT ROUTINE USED TO WATCH IF
7345 ;* WE'RE IN A LOOP TOO-LONG
7346 ;* CALL= JSR RS,$STOUT
7347 ;* ERROR X ;RETURNS HERE ON TIMEOUT
7348 ;* BR
7349 ;* ;RETURNS HERE NO ERROR
7350 ;*
7351
7352 040444 020537 040500 $STOUT: CMP RS,$SAD ;SAME ADDR?
7353 040450 001405 BEQ 1$
7354 040452 010537 040500 MOV RS,$SAD ;NO-SAVE THIS ADDR.
7355 040456 005037 00502 CLR $CNT ;CLR CNT AT ADDR.
7356 040462 000403 BR 2$
7357 040464 005237 040502 1$: INC $CNT ;OVERFLOW?

```

```

7358 040470 100402          BMI      3$          ; YES-ERROR RETURN
7359 040472 062705 000004 2$:      ADD      #4,R5      ; NO-NOV ERROR RETURN
7360 040476 000205          3$:      RTS      R5          ; RETURN.
7361
7362 040500 000000          $$AD:   .WORD   0          ; CONTAINS LOOP ADDR.
7363 040502 000000          $CNT:   .WORD   0          ; # OF TIMES AT ADDR.
7364
7365
7366
7367
7368
7369
7370
7371
7372
7373
7374 040504 000005          $RESET: RESET          ; RESET THE WORLD.
7375
7376
7377 040516 005737 037532  : *      MOV      @2$,1$      ; /READ DEVICE REG 2$, PUT DATA IN 1$.
7378 040522 001004          TST      $AERR          ; IF NO ERROR, LOOP
7379 040524 062737 000002 040540 BNE      10$          ; THERE WAS AN ERROR.
7380
7381
7382
7383
7384 040532 000764          ADD     #2,2$          ; UPDATE DEVICE ADDR.
7385
7386
7387
7388
7389
7390
7391
7392
7393
7394
7395
7396
7397
7398
7399
7400
7401
7402
7403 040542 005737 040624          BR      $RESET          ; YOU SEE, WE HAVE TO PROTECT OUR SELF!
7404 040542 100016          10$:    BR      $RESET          ; IF 2$ CONTAINED A VALID ADDR, WE
7405 040546 012737 000002 040614          1$:    BR      $RESET          ; MUST KEEP TRYING UNTIL WE GENERATE
7406 040550 052777 000115 000040          2$:    BR      $RESET          ; AN INVALID ADDR.
7407 040556 005037 177776          ; JUNK LOC.
7408 040564 005737 040614          ; DUMB ADDR. FORCES INIT OF DMC/KMC.
7409 040570 001375
7410 040574 005077 000022
7411

```

; * THIS ROUTINE REPLACES WHAT THE USER WOULD ORDINARILY
 ; * USE FOR A RESET. FIRST, WE DO A RESET INSTRUCTION.
 ; * THEN WE CLR ".DVLST" WHICH FORCES US TO RESET BOTH THE
 ; * KMC AND DMC AS SOON AS A DEVICE IS REFERENCED.
 ; *
 ; * CALL=JSR PC,\$RESET ; REPLACES "RESET INSTRUCTION
 ; * ; RETURNS HERE.
 ; *
 ; SDELAY- ROUTINE TO GIVE A MINOR DELAY.
 ; IS NOT TIME DEPENDENT CODE SENCE
 ; NOT USED TO GET SPECIFIC TIME BUT
 ; JUST A LITTLE DELAY.
 ;
 ; THAT IS UNLESS A REAL TIME CLOCK IS PRESENT!
 ; THEN WE'LL GENERATE A TIME BETWEEN 16MS TO 32 MS
 ;
 ; CALL= JSR PC, SDELAY

```

SDELAY:
7403 040542 005737 040624          TST      RTCCSR          ; CLOCK PRESENT?
7404 040542 100016          BPL      10$
7405 040546 012737 000002 040614          MOV      #2,TIME
7406 040550 052777 000115 000040          BIS      #115,@RTCCSR ; START CLOCK
7407 040556 005037 177776          CLR     PS
7408 040564 005737 040614          1$:    TST      TIME
7409 040570 001375          BNE     1$
7410 040574 005077 000022          CLR     @RTCCSR          ; STOP CLOCK
7411

```

```

7412
7413 040602 000207          RTS PC
7414 040604 105237 740614 10$: INCB TIME
7415 040610 001375          BNE 10$
7416 040612 000207          RTS PC
7417
7418 040614 000000          TIME: .WORD 0
7419
7420 040616 005337 040614 CLKINT: DEC TIME
7421 040622 000002          RTI
7422 040624 000000          RTCCSR: .WORD 0 ;CLOCK CSR IF USED.
7423
7424 ;SRESET WILL RESET WHOLE LPA SYSTEM. WITH USER MICRO-CODE.
7425 ;CALL= JSR PC,SRESET
7426
7427 040626 052777 040000 140632 SRESET: BIS #BIT14,@KMADO ;SET INIT BIT
7428 040634 012700 000144 MOV #100,RO
7429 040640 004737 040542 1$: JSR PC,$DELAY ;DELAY
7430 040644 005300 DEC RO
7431 040646 001374 BNE 1$
7432 040650 012737 044120 040700 MOV #MMAST,10$
7433 040656 122737 000115 001772 CMPB #'M,$VERSN
7434 040664 001403 BEQ 9$
7435 040666 012737 050124 040700 MOV #DMAST,10$
7436 040674 004537 037534 9$: JSR R5,$LOAD ;RELOAD MICRO CODE.
7437 040700 044120 10$: .WORD MMAST ;CURREN VERSION.
7438
7439
7440 040702 012777 104000 140556 MOV #BIT15!BIT11,@KMADO ;START KMC.
7441 040710 012700 000005 MOV #5,RO
7442 040714 004737 040542 2$: JSR PC,$DELAY
7443 040720 005300 DEC RO
7444 040722 001374 BNE 2$
7445 040724 005077 140542 CLR @KMAD2
7446 040730 005077 140542 CLR @KMAD4
7447 040734 000207 RTS PC
7448
7449 ;*
7450 ;*THIS MACRO ALLOWS THE OPERATOR TO TALK TO
7451 ;*ANY DEVICE ON THE I/O BUS
7452 ;*USER MUST START AT THIS ADDR.
7453 ;*HE MUST SAY EITHER "E" FOR EXAMINE, OR "D" FOR DEPOSIT.
7454 ;*"E" IS DEFAULT.
7455 ;*NEXT, HE MUST SUPPLY AN ADDR.
7456 ;*NOTE IF ADDR. IS NOT FOUND ON I/O BUS, A HALT
7457 ;*WILL OCCUR.
7458
7458 040736 $UTK: CLR .DVLS
7459 040736 005037 001514 21$: BR 65$
7460 040742 104401 040750 ;:65$: .ASCIZ <200>#E OR D?# ;:TYPE ASCIZ STRING
7461 040742 000405 ;:64$: .ASCIZ ;:GET OVER THE ASCIZ
7462
7463
7464 040762 105777 140156 1$: TSTB @$TKS
7465

```

7466	040766	100375			BPL	1\$		
7467	040770	117737	140152	041112	MOVB	2\$TKB,20\$;GET INPUT
7468	040776	104401	041112		TYPE	20\$;ECHO, NEXT MESSAGE.
7469	041002	142737	000240	041112	BICB	#240,20\$;STRIP PARITY, LC
7470	041010	104412			RDOCT			;GET ADDR.
7471	041012	012637	041110		MOV	(SP)+,14\$		
7472	041016	123727	041112	000104	CMPB	20\$,#1'D		;DEPOSIT?
7473	041024	001411			BEG	10\$		
7474								
7475	041026	004537	340350		JSR	R5,\$INLP		;GET DATA
7476	041032	041110			.WORD	14\$		
7477	041034	041046			.WORD	5\$		
7478								
7479	041036	013746	041046		MOV	5\$,-(SP)		::SAVE 5\$ FOR TYPEOUT
7480	041042	104402			TYPOC			::GO TYPE--OCTAL ASCII(ALL DIGITS)
7481	041044	000736			BR	21\$::LOOP.
7482	041046	000000			.WORD	0		
7483								
7484	041050							
7485	041050	104401	041056		TYPE	67\$::TYPE ASCIZ STRING
7486	041054	000404			BR	66\$::GET OVER THE ASCIZ
7487					.ASCIZ	<200>#DATA= #		
7488	041066							
7489	041066	104412			RDOCT			
7490	041070	012637	041106		MOV	(SP)+,13\$		
7491								
7492	041074	004537	040252		JSR	R5,\$OUTLP		;OUTPUT ROUTINE.
7493	041100	041110			.WORD	14\$;DEVICE ADDR.
7494	041102	041106			.WORD	13\$;DATA
7495	041104	000716			BR	21\$		
7496								
7497	041106	000000			.WORD	0		
7498	041110	000000			.WORD	0		
7499	041112	100001	042504	044526	.ASCIZ	<1><200>#DEVICE ADDR= #		
7500	041120	042503	040440	042104				
7501	041126	036522	000040					

.EVEN

```

: THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
: IF UNFOUND GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
: TO SET UP THE USER PROGRAM TO LINK TO FILE "DRLPX2" FOR
: SAMPLE TAKEING PURPOSES.
: TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
: A/D CSR IN BSEL 4 AND 5.
: (2) HE MUST CALL THIS ROUTINE:
:     JSR     R5,$PUTS      ;CALL SET UP ROUTINE.
:     .WORD  ADCSR        ;ADDR. OF A/D CSR.
:     ;RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
:     ;(UNTILL ONE DOES A RESET)
:
: (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
    
```

7502
7503
7504
7505
7506
7507
7508
7509
7510
7511
7512
7513
7514
7515
7516
7517
7518
7519

START CONVERSION CAUTION*DO WITH MOVB INSTR.!
(4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONE)
(5)READ KMC REG 4,5 FOR A/D RESULT.
(6) TO TAKE MORE SAMPLES, SIMPLY PUT A/D CSR INTO
BSEL 4,5 AND CODE 6 INTO BSEL 2.

7520
7521
7522
7523
7524
7525
7526 041132 012537 041142
7527 041136 004537 040350
7528 041142 000000
7529 041144 041240
7530 041146 113777 040412 140326
7531 041154 113777 040412 140322
7532 041162 013737 041142 041202
7533 041170 062737 000002 041202
7534 041176 004537 040350
7535 041202 000000
7536 041204 041240
7537 041206 113777 040412 140260
7538 041214 152777 000340 140260
7539 041222 152777 000300 140254
7540 041230 152777 000300 140236
7541 041236 000205
7542 041240 000000
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552 041242 010046
7553 041244 016600 000002
7554 041250 005740
7555 041252 111000
7556 041254 006300
7557 041256 016000 041276
7558 041262 000200
7559
7560
7561
7562
7563 041264 011646
7564 041266 016666 000004 000002
7565 041274 000002
7566
7567
7568
7569
7570
7571
7572
7573

```
$PUTS: MOV (5)+,1$ ;GET ADDR OF ADDR OF A/D
        JSR R5,$INLP
1$:      .WORD 0
        .WORD 10$
        MOVB $OFS,@KMA06
        MOVB $OFS,@KMA07
        MOV 1$,2$
        ADD #2,2$
        JSR R5,$INLP
2$:      .WORD 0
        .WORD 10$
        MOVB $OFS,@KMA03
        BISB #340,@KMA06
        BISB #300,@KMA07
        BISB #300,@KMA03
        RTS R5
10$:     .WORD 0

.SBTTL TRAP DEC0DER

;*****
;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
;GO TO THAT ROUTINE.

$TRAP:  MOV RO,-(SP) ;SAVE RO
        MOV 2(SP),RO ;GET TRAP ADDRESS
        TST -(RO) ;BACKUP BY 2
        MOVB (RO),RO ;GET RIGHT BYTE OF TRAP
        ASL RO ;POSITION FOR INDEXING
        MOV $TRPAD(RO),RO ;INDEX TO TABLE
        RTS RO ;GO TO ROUTINE

; ;THIS IS USE TO HANDLE THE "GETPRI" MACRO

$TRAP2: MOV (SP),-(SP) ;MOVE THE PC DOWN
        MOV 4(SP),2(SP) ;MOVE THE PSW DOWN
        RTI ;RESTORE THE PSW

.SBTTL TRAP TABLE

;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
;BY THE "TRAP" INSTRUCTION.

: ROUTINE
: -----
:
```

7574	041276	041264	\$TRPAD:	.WORD	\$TRAP2		
7575	041300	033052		\$TYPE	::CALL=TYPE	TRAP+1(104401)	TTY TYPEOUT ROUTINE
7576	041302	031036		\$TYPOC	::CALL=TYPOC	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
7577	041304	031012		\$TYPOS	::CALL=TYPOS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
7578	041306	031052		\$TYPON	::CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
7579	041310	031240		\$TYPDS	::CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
7580							
7581	041312	032346		\$GTSWR	::CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
7582							
7583	041314	032276		\$CKSWR	::CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
7584	041316	032560		\$RDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
7585	041320	032700		\$RDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
7586	041322	033334		\$RDOCT	::CALL=RDOCT	TRAP+12(104412)	READ AN OCTAL NUMBER FROM TTY
7587							
7588							
7589							
7590	041324		RBUFR:				
7591	041324	104401		TYPE	65\$::TYPE ASCIZ STRING
7592	041330	000410	041332	BR	64\$::GET OVER THE ASCIZ
7593				::65\$:	.ASCIZ	<200>#BUFFER START? *	
7594	041352			64\$:			
7595	041352	104412		RDOCT			
7596	041354	012600		MOV	(SP)+,RO		
7597	041356	001002		BNE	1\$		
7598	041360	012700	056246	MOV	#BUFFER,RO		
7599	041364			1\$:			
7600	041364	104401	041372	TYPE	67\$::TYPE ASCIZ STRING
7601	041370	000406		BR	66\$::GET OVER THE ASCIZ
7602				::67\$:	.ASCIZ	#CHAN INC? *	
7603	041406			66\$:			
7604	041406	104412		RDOCT			
7605	041410	012601		MOV	(SP)+,R1		
7606	041412	001002		BNE	2\$		
7607	041414	012701	000020	MOV	#16.,R1		
7608	041420	006301		2\$:	ASL R1		
7609	041422	104401	041430	TYPE	69\$::TYPE ASCIZ STRING
7610	041426	000407		BR	68\$::GET OVER THE ASCIZ
7611				::69\$:	.ASCIZ	<200>#ADDR. DATA*	
7612	041446			68\$:			
7613	041446			3\$:			
7614	041446	104401	041454	TYPE	71\$::TYPE ASCIZ STRING
7615	041452	000401		BR	70\$::GET OVER THE ASCIZ
7616				::71\$:	.ASCIZ	<200>##	
7617	041456			70\$:			
7618	041456	010046		MOV	RO,-(SP)		::SAVE RO FOR TYPEOUT
7619	041460	104402		TYPOC			::GO TYPE--OCTAL ASCII(ALL DIGITS)
7620	041462	104401	041470	TYPE	73\$::TYPE ASCIZ STRING
7621	041466	000402		BR	72\$::GET OVER THE ASCIZ
7622				::73\$:	.ASCIZ	* *	
7623	041474			72\$:			
7624	041474	011046		MOV	(0),-(SP)		::SAVE (0) FOR TYPEOUT
7625	041476	104402		TYPOC			::GO TYPE--OCTAL ASCII(ALL DIGITS)
7626	041500	060100		ADD R1,RO			
7627	041502	000761		BR	3\$		

Address	Offset	PC	PC+4	PC+8	Message
7628					
7629					
7630	041504	046200	040520	040440	EM1: .SBTTL .ASCIZ MESSAGES. .ASCIZ <200>#LPA ADDRESS ERROR#
7631	041512	042104	042522	051523	
7632	041520	042440	051122	051117	
7633	041526	000			
7634					
7635	041527	200	050114	020101	EM2: .ASCIZ <200>#LPA (KMC-11) DATA ERROR#
7636	041534	045450	041515	030455	
7637	041542	024461	042040	052101	
7638	041550	020101	051105	047522	
7639	041556	000122			
7640					
7641	041560	046200	040520	024040	EM3: .ASCIZ <200>#LPA (KMC-11) INSTRUCTION ERROR#
7642	041566	046513	026503	030461	
7643	041574	020051	047111	052123	
7644	041602	052522	052103	047511	
7645	041610	020116	051105	047522	
7646	041616	000122			
7647					
7648	041620	046200	040520	024040	EM4: .ASCIZ <200>#LPA (M8254) INIT ERROR#
7649	041626	034115	032462	024464	
7650	041634	044440	044516	020124	
7651	041642	051105	047522	000122	
7652					
7653	041650	046200	040520	024040	EM6: .ASCIZ <200>#LPA (M8254) SILO DATA ERROR#
7654	041656	034115	032462	024464	
7655	041664	051440	046111	020117	
7656	041672	040504	040524	042440	
7657	041700	051122	051117	000	
7658					
7659	041705	200	050114	020101	EM7: .ASCIZ <200>#LPA (M8254) FAST PATH DATA ERROR#
7660	041712	046450	031070	032065	
7661	041720	020051	040506	052123	
7662	041726	050040	052101	020110	
7663	041734	040504	040524	042440	
7664	041742	051122	051117	000	
7665					
7666	041747	200	050114	020101	EM10: .ASCIZ <200>#LPA (AD11K) CSR ERROR#
7667	041754	040450	030504	045461	
7668	041762	020051	051503	020122	
7669	041770	051105	047522	000122	
7670					
7671	041776	046200	040520	024040	EM11: .ASCIZ <200>#LPA (KW11K) CSR ERROR#
7672	042004	053513	030461	024513	
7673	042012	041440	051123	042440	
7674	042020	051122	051117	000	
7675					
7676	042025	200	050114	020101	EM12: .ASCIZ <200>#LPA (DR11K) ERROR#
7677	042032	042050	030522	045461	
7678	042040	020051	051105	047522	
7679	042046	000122			
7680					
7681	042050	046200	040520	024040	EM14: .ASCIZ <200>#LPA (AA-11K) ERROR#

7682	042056	040501	030455	045461															
7683	042064	020051	051105	047522															
7684	042072	000122																	
7685																			
7686	042074	046200	040520	024040	EM15:	.ASCIZ	<200>	#LPA	(AR11)	ERROR#									
7687	042102	051101	030461	020051															
7688	042110	051105	047522	000122															
7689																			
7690	042116	046200	040520	024040	EM16:	.ASCIZ	<200>	#LPA	(LPS-11)	ERROR#									
7691	042124	050114	026523	030461															
7692	042132	020051	051105	047522															
7693	042140	000122																	
7694																			
7695	042142	046200	040520	030455	EM17:	.ASCIZ	<200>	#LPA-11	FUNCTIONAL	ERROR#									
7696	042150	020061	052506	041516															
7697	042156	044524	047117	046101															
7698	042164	042440	051122	051117															
7699	042172	000																	
7700																			
7701	042173	200	042522	042520	EM20:	.ASCIZ	<200>	#REPEATIBILITY	OR CHAN	AVERAGE	ERROR#								
7702	042200	052101	041111	046111															
7703	042206	052111	020131	051117															
7704	042214	041440	040510	020116															
7705	042222	053101	051105	043501															
7706	042230	020105	051105	047522															
7707	042236	000122																	
7708																			
7709	042240	042600	051122	041520	DH1:	.ASCIZ	<200>	#ERRPC	ADDRESS#										
7710	042246	020040	040440	042104															
7711	042254	042522	051523	000															
7712																			
7713	042261	200	051105	050122	DH2:	.ASCIZ	<200>	#ERRPC	EXP'ED	REC'ED#									
7714	042265	020103	020040	054105															
7715	042274	023520	042105	020040															
7716	042302	051040	041505	042447															
7717	042310	000104																	
7718																			
7719	042312	042600	051122	041520	DH3:	.ASCIZ	<200>	#ERRPC#											
7720	042320	000																	
7721	042321	200	051105	050122	DH17:	.ASCIZ	<200>	#ERRPC	TESTNO	EXP'ED	REC'ED#								
7722	042326	020103	020040	042524															
7723	042334	052123	047516	020040															
7724	042342	054105	023520	042105															
7725	042350	020040	042522	023503															
7726	042356	042105	000																
7727																			
7728	042361	200	042524	052123	DH20:	.ASCIZ	<200>	#TEST	S/B	CHAN	HIGH	LOW#							
7729	042366	020040	020040	027523															
7730	042374	020102	020040	020040															
7731	042402	041440	040510	020116															
7732	042410	020040	044040	043511															
7733	042416	020110	020040	046040															
7734	042424	053517	000																
7735	042427	200	051105	050122	DH21:	.ASCIZ	<200>	#ERRPC	TSTNO	USJNO	USRDA	ALPCO	ALPCI	ALPSO#					

7736	042434	020103	020040	051524			
7737	042442	047124	020117	020040			
7738	042450	051525	047112	020117			
7739	042456	020040	051525	042122			
7740	042464	020101	020040	046101			
7741	042472	041520	020117	020040			
7742	042500	046101	041520	020111			
7743	042506	020040	046101	051520			
7744	042514	000117					
7745							
7746						.EVEN	
7747	042516	001116	001466	000000	DT1:	.WORD	\$ERRPC, KMADO, 0
7748							
7749	042524	001116	001124	001126	DT2:	.WORD	\$ERRPC, \$GDDAT, \$BDDAT, 0
7750	042532	000000					
7751							
7752	042534	001116	000000		DT3:	.WORD	\$ERRPC, 0
7753	042540	001116	001200	001124	DT17:	.WORD	\$ERRPC, \$TESTN, \$GDDAT, \$BDDAT, 0
7754	042546	001126	000000				
7755							
7756	042552	001200	034514	034466	DT20:		\$TESTN, AVEXP, CHAN, RHIGH, RLOW, 0
7757	042560	034522	034520	000000			
7758							
7759	042566	001116	001200	001756	DT21:		\$ERRPC, \$TESTN, USJNO, ALPADL, ALPCO, ALPCI, ALPSO, 0
7760	042574	001770	001764	001762			
7761	042602	001766	000000				
7762	042606	000000	000000		DF0:	.WORD	0, 0
7763							
7764	042612	020040	000		M2SP:	.ASCIZ	* *
7765							
7766							
7767	042615	200	040515	042513	M1:	.ASCII	<200>"MAKE YOUR OWN JOB(S) ROUTINE FOR LPA-11"
7768	042622	054440	052517	020122			
7769	042630	053517	020116	047512			
7770	042636	024102	024523	051040			
7771	042644	052517	044524	042516			
7772	042652	043040	051117	046040			
7773	042660	040520	030455	061			
7774	042665	200	046120	040505			
7775	042672	042523	040440	051516	.ASCII	<200>"PLEASE ANSWER ALL QUESTIONS, IF IN DOUBT, REFERR TO DOCUMENTATION"	
7776	042700	042527	020122	046101			
7777	042706	020114	052521	051505			
7778	042714	044524	047117	026123			
7779	042722	044440	020106	047111			
7780	042730	042040	052517	052102			
7781	042736	020054	042522	042506			
7782	042744	051122	052040	020117			
7783	042752	047504	052503	042515			
7784	042760	052116	052101	047511			
7785	042766	116					
7786	042767	200	042200	042105	.ASCIZ	<200><200>"DEDICTED OR MULTIUSER (D OR M)"	
7787	042774	041511	042524	020104			
7788	043002	051117	046440	046125			
7789	043010	044524	051525	051105			

7790	043016	024040	020104	051117		
7791	043024	046440	000051			
7792						
7793	043030	051600	052105	041440	M2:	.ASCIZ <200>"SET CLOCK CSR= "
7794	043036	047514	045503	041440		
7795	043044	051123	020075	000		
7796						
7797	043051	200	042523	020124	M3:	.ASCIZ <200>"SET PRESENT BUFFER (2'S COMP) TO "
7798	043056	051120	051505	047105		
7799	043064	020124	052502	043106		
7800	043072	051105	024040	023462		
7801	043100	020123	047503	050115		
7802	043106	020051	047524	000040		
7803						
7804	043114	044200	053517	046440	M4:	.ASCIZ <200>"HOW MANY JOBS "
7805	043122	047101	020131	047512		
7806	043130	051502	000040			
7807						
7808	043134	045200	041117	021440	M5:	.ASCIZ <200>"JOB # "
7809	043142	000040				
7810	043144	051440	040524	051524	M6:	.ASCIZ " STATS:"
7811	043152	000072				
7812	043154	044600	047457	042040	M7:	.ASCIZ <200>"I/O DEVICE START MODE WORD= "
7813	043162	053105	041511	020105		
7814	043170	052123	051101	020124		
7815	043176	047515	042504	053440		
7816	043204	051117	036504	000040		
7817	043212	041200	043125	042506	M8:	.ASCIZ <200>"BUFFER SIZE (2'S COMP)= "
7818	043220	020122	044523	042532		
7819	043226	024040	023462	020123		
7820	043234	047503	050115	036451		
7821	043242	000040				
7822	043244	052600	042523	020122	M9:	.ASCIZ <200>"USER STATUS WORD ADDR= "
7823	043252	052123	052101	051525		
7824	043260	053440	051117	020104		
7825	043266	042101	051104	020075		
7826	043274	000				
7827	043275	200	047510	020127	M97:	.ASCII <200>"HOW MANY TIMES TO FILL BUFFERS (NORM =1) "
7828	043302	040515	054516	052040		
7829	043310	046511	051505	052040		
7830	043316	020117	044506	046114		
7831	043324	041040	043125	042506		
7832	043332	051522	024040	047516		
7833	043340	046522	036440	024461		
7834	043346	040				
7835	043347	200	047510	020127	M10:	.ASCIZ <200>"HOW MANY BUFFERS ? "
7836	043354	040515	054516	041040		
7837	043362	043125	042506	051522		
7838	043370	037440	000040			
7839	043374	044600	020123	042504	M11:	.ASCIZ <200>"IS DEVICE OVERRUN FATAL?"
7840	043402	044526	042503	047440		
7841	043410	042526	051122	047125		
7842	043416	043040	052101	046101		
7843	043424	000077				

7844	043426	044600	020123	052502	M12:	.ASCIZ	<200>"IS BUFFER OVERRUN FATAL?"
7845	043434	043106	051105	047440			
7846	043442	042526	051122	047125			
7847	043450	043040	052101	046101			
7848	043456	000077					
7849	043460	044600	051516	043125	M13:	.ASCIZ	<200>"INSUFFICIENT CORE FOR BUFFERS-RETYPE THIS JOBS PRAMETERS?"
7850	043466	044506	047105	020124			
7851	043474	047503	042522	043040			
7852	043502	051117	041040	043125			
7853	043510	042506	051522	051055			
7854	043516	052105	050131	020105			
7855	043524	044124	051511	045040			
7856	043532	041117	020123	051120			
7857	043540	046501	052105	051105			
7858	043546	037523	000				
7859	043551	200	042504	040514	M14:	.ASCIZ	<200>"DELAY BEFORE START"
7860	043556	020131	042502	047506			
7861	043564	042522	051440	040524			
7862	043572	052122	000				
7863	043575	200	052516	041115	M15:	.ASCIZ	<200>"NUMBER OF CHNNELS?"
7864	043602	051105	047440	020106			
7865	043610	044103	047116	046105			
7866	043616	037523	000				
7867	043621	200	044514	052123	M16:	.ASCIZ	<200>"LIST RANDOM CHANNELS"
7868	043626	051040	047101	047504			
7869	043634	020115	044103	047101			
7870	043642	042516	051514	000			
7871	043647	200	040523	050115	M17:	.ASCIZ	<200>"SAMPLE#"
7872	043654	042514	000043				
7873	043660	045600	045501	020105	M18:	.ASCIZ	<200>"MAKE BUFFER NOW?"
7874	043666	052502	043106	051105			
7875	043674	047040	053517	000077			
7876	043702	047200	053505	047440	M0:	.ASCIZ	<200>"NEW OR OLD DATA?"
7877	043710	020122	046117	020104			
7878	043716	040504	040524	000077			
7879	043724	050200	042522	051523	M19:	.ASCIZ	<200>"PRESS CONTINUE WHEN READY"
7880	043732	041440	047117	044524			
7881	043740	052516	020105	044127			
7882	043746	047105	051040	040505			
7883	043754	054504	000				
7884	043757	200	044103	047101	M20:	.ASCIZ	<200>"CHAN ADDRESS WORD?"
7885	043764	040440	042104	042522			
7886	043772	051523	053440	051117			
7887	044000	037504	000				
7888	044003	200	046103	041517	M21:	.ASCIZ	<200>"CLOCK RATE?"
7889	044010	020113	040522	042524			
7890	044016	000077					
7891	044020	051600	040524	052122	M22:	.ASCIZ	<200>"START/EVENT MARK WORD? "
7892	044026	042457	042526	052116			
7893	044034	046440	051101	020113			
7894	044042	047527	042122	020077			
7895	044050	000					
7896	044051	200	052123	051101	M23:	.ASCIZ	<200>"START MASK? "
7897	044056	020124	040515	045523			

MAINDEC -11- DRLPA-A
DRLPA.P11

MACY11 27(654) 13-DEC-77 12:58 PAGE 167
.ASCIZ MESSAGES.

SEQ 0197

7898	044064	020077	000				
7899	044067	200	053105	047105	M24:	.ASCIZ	<200>"EVENT MARK MASK? "
7900	044074	020124	040515	045522			
7901	044102	046440	051501	037513			
7902	044110	000040					
7903							
7904					.EVEN		

7905
7906
7907
7908
7909 044112 000000
7910 044114 000000
7911 044116 000000
7912
7913
7914 054130 000000
7915 054132 000012
7916 054156
7917
7918
7919
7920
7921
7922
7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947 000022
7948 000100
7949 000000
7950 000000
7951 000120
7952 000035
7953 000000
7954 000235
7955 000002
7956 000015
7957 000040
7958 000040

```

; ** REQUEST DESCRIPTOR ARRAY FOR THE KW-11K CLOCK
; ** THIS TABLE WILL BE ALTERED BY THIS PROGRAM
; ** AND BY THE LPA-11 OPTION.
KWT:  .WORD 0 ;CLOCK START MODE WORD.
      .WORD 0 ;CLOCK PRESET REGISTER.
      .WORD 0 ;CLOCK BUFFER REGISTER.

DEVLST: .WORD 0 ;MODE WORD FOR START.
        .BLKW 10. ;TEN ADDRESSES ON START.

DMDT:  .TITLE DMDT
       .SBTTL DMDT -- DEDICATED MODE DISPATCH TABLE
       .IDENT /LPA.03/

```

```

; COPYRIGHT 1976, 1977 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR USE
; ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION
; OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM,
; EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.
; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT
; NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
; EQUIPMENT CORPORATION.
; DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

```

```

; CHARLES A. SAMUELSON
; FEBRUARY 4, 1977

```

```

+
; DMDT -- DEDICATED MODE DISPATCH TABLE
; TABLE FOR LPA11 MICRO PROCESSOR DEDICATED MODE SAMPLING
-

```

```

; DEFINED VALUES
DMDSIZ=18. ;LENGTH OF DMDT BUFFER IN BYTES
SDT=100 ;SLAVE DISPATCH TABLE START ADDRESS
CAINC=0 ;CHANNEL ADDRESS INCREMENT VALUE
AD1SR=0 ;ADC #1 STATUS REGISTER ADDRESS LOW BYTE
SEX=120 ;SELECT EXTERNAL CLOCK START AND INTERRUPT ENABLE
RONPR=35 ;REQUEST OUTPUT NPR IN MICRO-PROCESSOR
CLR=0 ;CLEAR AD STATUS REGISTER
RONPRL=235 ;REQUEST OUTPUT NPR LOW BYTE IN MICRO-PROCESSOR
AD1DL=2 ;ADC #1 DATA REGISTER ADDRESS LOW BYTE
RINPR=15 ;REQUEST INPUT NPR
SCS=40 ;SELECT CLOCK OVERFLOW START FOR ADC'S
AD2SR=40 ;ADC #2 STATUS REGISTER ADDRESS LOW BYTE

```

7959	000042				AD2DRL=42	:ADC #2 DATA REGISTER ADDRESS LOW BYTE
7960	000001				AD1SRH=1	:ADC #1 STATUS REGISTER HIGH BYTE ADDRESS
7961	000041				AD2SRH=41	:ADC #2 STATUS REGISTER HIGH BYTE ADDRESS
7962	000020				SEN=20	:SELECT EXTERNAL START, NO INTERRUPT ENABLE
7963					:	
7964					:	
7965	054156				DMDT::	
7966	054156				D.OES::	:ONE ADC, EXTERNAL TRIGGER, SINGLE CHANNEL
7967	054156	162	000	120	.BYTE	DMDSIZ+<3*40>,AD1SRL,SEX,RONPR,AD1DRL,RINPR,CLR,RONPRL,SDT
7968	054161	035	002	015		
7969	054164	000	235	100		
7970		054201			.=D.OES+23	
7971	054201				D.OEQ::	:ONE ADC, EXTERNAL TRIGGER, SEQUENTIAL CHANNEL
7972	054201	162	000	000	.BYTE	DMDSIZ+<3*40>,CAINC,AD1SRL,SEX,RONPR,AD1DRL,RINPR,CLR,RONPRL,SDT
7973	054204	120	035	002		
7974	054207	015	000	235		
7975	054212	100				
7976	054213	000			.BYTE	CAINC
7977		054224			.=D.OEQ+23	
7978	054224				D.OCS::	:ONE ADC, CLOCK TRIGGER, SINGLE CHANNEL
7979	054224	162	000	040	.BYTE	DMDSIZ+<3*40>,AD1SRL,SCS,RONPR,AD1DRL,RINPR,CLR,RONPRL,SDT
7980	054227	035	002	015		
7981	054232	000	235	100		
7982		054247			.=D.OCS+23	
7983	054247				D.OCQ::	:ONE ADC, CLOCK TRIGGER, SEQUENTIAL CHANNEL
7984	054247	162	000	000	.BYTE	DMDSIZ+<3*40>,CAINC,AD1SRL,SCS,RONPR,AD1DRL,RINPR,CLR,RONPRL,SDT
7985	054252	040	035	002		
7986	054255	015	000	235		
7987	054260	100				
7988	054261	000			.BYTE	CAINC
7989		054272			.=D.OCQ+23	
7990	054272				D.TES::	:TWO ADC, EXTERNAL TRIGGER, SINGLE CHANNEL
7991	054272	162	000	120	.BYTE	DMDSIZ+<3*40>,AD1SRL,SEX,RONPR,AD1DRL,RINPR,CLR,RONPRL,SDT+10
7992	054275	035	002	015		
7993	054300	000	235	110		
7994	054303	040	120	035	.BYTE	AD2SRL,SEX,RONPR,AD2DRL,RINPR,CLR,RONPRL,SDT
7995	054306	042	015	000		
7996	054311	235	100			
7997		054315			.=D.TES+23	
7998	054315				D.TEQ::	:TWO ADC, EXTERNAL TRIGGER, SEQUENTIAL CHANNEL
7999	054315	162	000	000	.BYTE	DMDSIZ+<3*40>,CAINC,AD1SRL,SEX,RONPR,AD1DRL,RINPR,CLR,RONPRL
8000	054320	120	035	002		
8001	054323	015	000	235		
8002	054326	111	000	040	.BYTE	SDT+11,CAINC,AD2SRL,SEX,RONPR,AD2DRL,RINPR,CLR,RONPRL,SDT
8003	054331	120	035	042		
8004	054334	015	000	235		
8005	054337	100				
8006		054340			.=D.TEQ+23	
8007	054340				D.TCS::	:TWO ADC, CLOCK TRIGGER, SINGLE CHANNEL
8008	054340	262	000	040	.BYTE	DMDSIZ+<5*40>,AD1SRL,SCS,RONPR,AD1DRL,RINPR,CLR,RONPRL
8009	054343	035	002	015		
8010	054346	000	235			
8011	054350	110	040	040	.BYTE	SDT+10,AD2SRL,SCS,RONPR,AD2DRL,RINPR,CLR,RONPRL,SDT
8012	054353	035	042	015		

8013	054356	000	235	100	
8014		054363			.=D.TCS+23
8015	054363				D.TCQ::
8016	054363	262	000	000	:TWO ADC, CLOCK TRIGGER, SEQUENTIAL CHANNEL
8017	054366	040	035	002	.BYTE DMDSIZ+(<5*40>,CAINC,AD1SRL,SCS,RONPR,AD1DRL,RINPR,CLR,RONPRL
8018	054371	015	000	235	
8019	054374	111	000	040	.BYTE SDT+11,CAINC,AD2SRL,SCS,RONPR,AD2DRL,RINPR,CLR,RONPRL,SDT
8020	054377	040	035	042	
8021	054402	015	000	235	
8022	054405	100			
8023		054406			.=D.TCQ+23
8024					:
8025					PARALLEL MODE TABLE
8026					:
8027	054406				D.TESP::
8028	054406	022	120	000	:TWO ADC, EXTERNAL TRIGGER, SINGLE, PARALLE
8029	054411	035	040	020	.BYTE DMDSIZ+(<0*40>,SEX,AD1SRL,RONPR,AD2SRL,SEN,RONPR,AD1DRL,RINPR
8030	054414	035	002	015	
8031	054417	042	015	001	.BYTE AD2DRL,RINPR,AD1SRH,RONPRL,AD2SRH,RONPRL,SDT+6
8032	054422	235	041	235	
8033	054425	106			
8034		054431			.=D.TESP+23
8035	054431				D.TEQP::
8036	054431	022	120	000	:TWO ADC, EXTERNAL TRIGGER, SEQUENTIAL, PARALLEL
8037	054434	035	040	020	.BYTE DMDSIZ+(<0*40>,SEX,AD1SRL,RONPR,AD2SRL,SEN,RONPR,CAINC,AD1DRL
8038	054437	035	000	002	
8039	054442	015	042	015	.BYTE RINPR,AD2DRL,RINPR,AD1SRH,RONPRL,AD2SRH,RONPRL,SDT+6
8040	054445	001	235	041	
8041	054450	235	106		
8042		054454			.=D.TEQP+23
8043	054454				D.TCSP::
8044	054454	222	040	000	:TWO ADC, CLOCK TRIGGER, SINGLE, PARALLEL
8045	054457	035	040	040	.BYTE DMDSIZ+(<4*40>,SCS,AD1SRL,RONPR,AD2SRL,SCS,RONPR,AD1DRL,RINPR
8046	054462	035	002	015	
8047	054465	042	015	001	.BYTE AD2DRL,RINPR,AD1SRH,RONPRL,AD2SRH,RONPRL,SDT+6
8048	054470	235	041	235	
8049	054473	106			
8050		054477			.=D.TCSP+23
8051	054477				D.TCQP::
8052	054477	222	040	000	:TWO ADC, CLOCK TRIGGER, SEQUENTIAL, PARALLEL
8053	054502	035	040	040	.BYTE DMDSIZ+(<4*40>,SCS,AD1SRL,RONPR,AD2SRL,SCS,RONPR,CAINC,AD1DRL
8054	054505	035	000	002	
8055	054510	015	042	015	.BYTE RINPR,AD2DRL,RINPR,AD1SRH,RONPRL,AD2SRH,RONPRL,SDT+6
8056	054513	001	235	041	
8057	054516	235	106		
8058	054520	000050			TODOQ: .BLKW 40.
8059	054640	000000			TODOC: .WORD 0
8060					
8061					:THE FOLLOWING TABLE IS USED BY FLASH TO INDEX THE JOBS BASED
8062					:ON THE INFO THE LPA RETURNS AS ASSIGNED JOB NUMBERS.
8063					:FLASH INDEXES THIS TABLE WITH THE JOB NUMBER TO STORE THE RDA
8064					:ADDR. WHEN CODE=0,ANY OTHER TIME TABLE IS INDEXED TO GET
8065					:THE RDA ADDR. OF A PARTICULAR JOB.
8066					

8121	055130	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8122	055131	000	.BYTE	0	;/UNUSED
8123	055132	000000	.WORD	0	;/BUFFER ADDRESS #1
8124	055134	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8125	055135	000	.BYTE	0	;/UNUSED
8126					
8127	055136	055166	.WORD	JOBGR	;/RCLR (ADDR OF LIST OF RANDOM CHAN)
8128	055140	000	.BYTE	0	;/RCL EXTENDED ADR. BITS
8129	055141	000	.BYTE	0	;/UNUSED.
8130					
8131	055142	000	.BYTE	0	;/CHANNEL START ADDRESS
8132	055143	000	.BYTE	0	;/NUMBER OF CHANNELS
8133					
8134					
8135	055144	000	.BYTE	0	;/CHANNEL INCREMENT
8136	055145	000	.BYTE	0	;/FATAL ERROR MASK
8137					
8138	055146	000000	.WORD	0	;/DELAY
8139	055150	000000	.WORD	0	;/SAMPLE RATE
8140					
8141	055152	000	.BYTE	0	;/STWD
8142	055153	000	.BYTE	0	;/EMWD EVENT MARK DIGITAL INT WD*
8143					
8144	055154	000000	.WORD	0	;/ST MSK
8145	055156	000000	.WORD	0	;/EM MSK EVENT MARK DIGITAL INPUT MASK.
8146					
8147					;/END REG. TABLE
8148	055160	000003	.WORD	03	;/CONTAINS STOP CODE FOR THIS JOB.
8149	055162	000000	.WORD	0	;/CONTAINS LOOP COUNT FOR JOB.(NO OF TIMES THUR BUFFER.
8150					
8151	055164	000000	.WORD	0	;/USW USER STATUS WORD
8152					
8153					
8154	055166	000007	.BLKW	7	
8155					
8156					
8157					/**
8158					/**
8159					REQUEST DESCRIPTOR ARRAY FOR JOB #1
8160					/**
8161					THIS TABLE WILL BE ALERED BY THIS PROGRAM
8162					AND BY THE LPA-11 OPTION.
8163					/**
8163	055204	000000	.WORD	0	;/MODE INFORMATION.
8164	055206	000000	.WORD	0	;/WORD COUNT
8165	055210	055302	.WORD	0+JOB1U	;/USW (USER STATUS WORD)
8166	055212	000	.BYTE	0	;/USW (EXTENDED ADDR. BITS)
8167	055213	000	.BYTE	0	;/VBM (VALID BUFFER MASK)
8168	055214	000000	.WORD	0	;/BUFFER ADDRESS #0
8169	055216	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8170	055217	000	.BYTE	0	;/UNUSED
8171	055220	000000	.WORD	0	;/BUFFER ADDRESS #1
8172	055222	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8173	055223	000	.BYTE	0	;/UNUSED
8174	055224	000000	.WORD	0	;/BUFFER ADDRESS #1

8175	055226	000	.BYTE	0	;	EXTENDED ADDRESS BITS
8176	055227	000	.BYTE	0	;	UNUSED
8177	055230	000000	.WORD	0	;	BUFFER ADDRESS #1
8178	055232	000	.BYTE	0	;	EXTENDED ADDRESS BITS
8179	055233	000	.BYTE	0	;	UNUSED
8180	055234	000000	.WORD	0	;	BUFFER ADDRESS #1
8181	055236	000	.BYTE	0	;	EXTENDED ADDRESS BITS
8182	055237	000	.BYTE	0	;	UNUSED
8183	055240	000000	.WORD	0	;	BUFFER ADDRESS #1
8184	055242	000	.BYTE	0	;	EXTENDED ADDRESS BITS
8185	055243	000	.BYTE	0	;	UNUSED
8186	055244	000000	.WORD	0	;	BUFFER ADDRESS #1
8187	055246	000	.BYTE	0	;	EXTENDED ADDRESS BITS
8188	055247	000	.BYTE	0	;	UNUSED
8189	055250	000000	.WORD	0	;	BUFFER ADDRESS #1
8190	055252	000	.BYTE	0	;	EXTENDED ADDRESS BITS
8191	055253	000	.BYTE	0	;	UNUSED
8192						
8193	055254	055304	.WORD	JOB1R	;	RCLR (ADDR OF LIST OF RANDOM CHAN)
8194	055256	000	.BYTE	0	;	RCL EXTENDED ADR. BITS
8195	055257	000	.BYTE	0	;	UNUSED.
8196						
8197	055260	000	.BYTE	0	;	CHANNEL START ADDRESS
8198	055261	000	.BYTE	0	;	NUMBER OF CHANNELS
8199						
8200						
8201	055262	000	.BYTE	0	;	CHANNEL INCREMENT
8202	055263	000	.BYTE	0	;	FATAL ERROR MASK
8203						
8204	055264	000000	.WORD	0	;	DELAY
8205	055266	000000	.WORD	0	;	SAMPLE RATE
8206						
8207	055270	000	.BYTE	0	;	STWD
8208	055271	000	.BYTE	0	;	EMWD EVENT MARK DIGITAL INT WD#
8209						
8210	055272	000000	.WORD	0	;	ST MSK
8211	055274	000000	.WORD	0	;	EM MSK EVENT MARK DIGITAL IPUT MASK.
8212						
8213						
8214	055276	000003	.WORD	03	;	CONTAINS STOP CODE FOR THIS JOB.
8215	055300	000000	.WORD	0	;	CONTAINS LOOP COUNT FOR JOB.(NO OF TIMES THUR BUFFER.
8216						
8217	055302	000000	.WORD	0	;	USW USER STATUS WORD
8218						
8219						
8220	055304	000007	.BLKW	7		
8221						
8222						
8223						
8224						
8225						
8226						
8227						
8228						

 *** REQUEST DESCRIPTOR ARRAY FOR JOB #2

 *** THIS TABLE WILL BE ALERED BY THIS PROGRAM
 *** AND BY THE LPA-11 OPTION.

Address	Word	Value	Field	Value	Description
8229	055322	000000	JOB2: .WORD	0	;/MODE INFORMATION.
8230	055324	000000	.WORD	0	;/WORD COUNT
8231	055326	055420	.WORD	0+JOB2U ;/USW (USER STATUS WORD)	;/USW (EXTENDED ADDR. BITS)
8232	055330	000	.BYTE	0	;/VBM (VALID BUFFER MASK)
8233	055331	000	.BYTE	0	;/BUFFER ADDRESS #0
8234	055332	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8235	055334	000	.BYTE	0	;/UNUSED
8236	055335	000	.BYTE	0	;/BUFFER ADDRESS #1
8237	055336	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8238	055340	000	.BYTE	0	;/UNUSED
8239	055341	000	.BYTE	0	;/BUFFER ADDRESS #1
8240	055342	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8241	055344	000	.BYTE	0	;/UNUSED
8242	055345	000	.BYTE	0	;/BUFFER ADDRESS #1
8243	055346	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8244	055350	000	.BYTE	0	;/UNUSED
8245	055351	000	.BYTE	0	;/BUFFER ADDRESS #1
8246	055352	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8247	055354	000	.BYTE	0	;/UNUSED
8248	055355	000	.BYTE	0	;/BUFFER ADDRESS #1
8249	055356	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8250	055360	000	.BYTE	0	;/UNUSED
8251	055361	000	.BYTE	0	;/BUFFER ADDRESS #1
8252	055362	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8253	055364	000	.BYTE	0	;/UNUSED
8254	055365	000	.BYTE	0	;/BUFFER ADDRESS #1
8255	055366	000000	.WORD	0	;/EXTENDED ADDRESS BITS
8256	055370	000	.BYTE	0	;/UNUSED
8257	055371	000	.BYTE	0	;/UNUSED
8258					
8259	055372	055422	.WORD	JOB2R ;/RCLR (ADDR OF LIST OF RANDOM CHAN)	;/RCL EXTENDED ADR. BITS
8260	055374	000	.BYTE	0	;/UNUSED.
8261	055375	000	.BYTE	0	;/UNUSED.
8262					
8263	055376	000	JOB2S: .BYTE	0	;/CHANNEL START ADDRESS
8264	055377	000	.BYTE	0	;/NUMBER OF CHANNELS
8265					
8266					
8267	055400	000	.BYTE	0	;/CHANNEL INCREMENT
8268	055401	000	.BYTE	0	;/FATAL ERROR MASK
8269					
8270	055402	000000	.WORD	0	;/DELAY
8271	055404	000000	.WORD	0	;/SAMPLE RATE
8272					
8273	055406	000	.BYTE	0	;/STWD
8274	055407	000	.BYTE	0	;/EMWD EVENT MARK DIGITAL INT WD#
8275					
8276	055410	000000	.WORD	0	;/ST MSK
8277	055412	000000	.WORD	0	;/EM MSK EVENT MARK DIGITAL INPUT MASK.
8278					
8279			;/END REG. TABLE		
8280	055414	000003	.WORD	03	;/CONTAINS STOP CODE FOR THIS JOB.
8281	055416	000000	.WORD	0	;/CONTAINS LOOP COUNT FOR JOB. (NO OF TIMES THUR BUFFER.
8282					

K16

DMDT MACY11 27(654) 13-DEC-77 12:58 PAGE 175
 DRLPA.P11 DMDT -- DEDICATED MODE DISPATCH TABLE

SEQ 0205

8283	055420	000000	JOB2U: .WORD	0	;/USW USER STATUS WORD
8284					
8285					
8286	055422	000007	JOB2R: .BLKW	7	
8287					
8288			;		
8289			;		
8290			;		
8291			;		
8292			;		
8293			;		
8294			;		
8295	055440	000000	JOB3: .WORD	0	;/MODE INFORMATION.
8296	055442	000000	.WORD	0	;/WORD COUNT
8297	055444	055536	.WORD	0+JOB3U	;/USW (USER STATUS WORD)
8298	055446	000	.BYTE	0	;/USW (EXTENDED ADDR. BITS)
8299	055447	000	.BYTE	0	;/VBM (VALID BUFFER MASK)
8300	055450	000000	.WORD	0	;/BUFFER ADDRESS #C
8301	055452	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8302	055453	000	.BYTE	0	;/UNUSED
8303	055454	000000	.WORD	0	;/BUFFER ADDRESS #1
8304	055456	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8305	055457	000	.BYTE	0	;/UNUSED
8306	055460	000000	.WORD	0	;/BUFFER ADDRESS #1
8307	055462	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8308	055463	000	.BYTE	0	;/UNUSED
8309	055464	000000	.WORD	0	;/BUFFER ADDRESS #1
8310	055466	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8311	055467	000	.BYTE	0	;/UNUSED
8312	055470	000000	.WORD	0	;/BUFFER ADDRESS #1
8313	055472	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8314	055473	000	.BYTE	0	;/UNUSED
8315	055474	000000	.WORD	0	;/BUFFER ADDRESS #1
8316	055476	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8317	055477	000	.BYTE	0	;/UNUSED
8318	055500	000000	.WORD	0	;/BUFFER ADDRESS #1
8319	055502	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8320	055503	000	.BYTE	0	;/UNUSED
8321	055504	000000	.WORD	0	;/BUFFER ADDRESS #1
8322	055506	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8323	055507	000	.BYTE	0	;/UNUSED
8324					
8325	055510	055540	.WORD	JOB3R	;/RCLR (ADDR OF LIST OF RANDOM CHAN)
8326	055512	000	.BYTE	0	;/RCL EXTENDED ADR. BITS
8327	055513	000	.BYTE	0	;/UNUSED.
8328					
8329	055514	000	JOB3S: .BYTE	0	;/CHANNEL START ADDRESS
8330	055515	000	.BYTE	0	;/NUMBER OF CHANNELS
8331					
8332					
8333	055516	000	.BYTE	0	;/CHANNEL INCREMENT
8334	055517	000	.BYTE	0	;/FATAL ERROR MASK
8335					
8336	055520	000000	.WORD	0	;/DELAY

```

8337 055522 000000 .WORD 0 ; /SAMPLE RATE
8338
8339 055524 000 .BYTE 0 ; /STWD
8340 055525 000 .BYTE 0 ; /EMWD EVENT MARK DIGITAL INT WD#
8341
8342 055526 000000 .WORD 0 ; /ST MSK
8343 055530 000000 .WORD 0 ; /EM MSK EVENT MARK DIGITAL IPUT MASK.
8344
8345 ;END REG. TABLE
8346 055532 000003 .WORD 03 ; /CONTAINS STOP CODE FOR THIS JOB.
8347 055534 000000 .WORD 0 ; /CONTAINS LOOP COUNT FOR JOB.(NO OF TIMES THUR BUFFER.
8348
8349 055536 000000 JOB3U: .WORD 0 ; /USW USER STATUS WORD
8350
8351
8352 055540 000007 JOB3R: .BLKW 7
8353
8354 ;**
8355 ;** REQUEST DESCRIPTOR ARRAY FOR JOB #4
8356 ;**
8357 ;** THIS TABLE WILL BE ALERED BY THIS PROGRAM
8358 ;** AND BY THE LPA-11 OPTION.
8359 ;**
8360
8361 055556 000000 JOB4: .WORD 0 ; /MODE INFORMATION.
8362 055560 000000 .WORD 0 ; /WORD COUNT
8363 055562 055654 .WORD 0+JOB4U ; /USW (USER STATUS WORD)
8364 055564 000 .BYTE 0 ; /USW (EXTENDED ADDR. BITS)
8365 055565 000 .BYTE 0 ; /VBM (VALID BUFFER MASK)
8366 055566 000000 .WORD 0 ; /BUFFER ADDRESS #0
8367 055570 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8368 055571 000 .BYTE 0 ; /UNUSED
8369 055572 000000 .WORD 0 ; /BUFFER ADDRESS #1
8370 055574 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8371 055575 000 .BYTE 0 ; /UNUSED
8372 055576 000000 .WORD 0 ; /BUFFER ADDRESS #1
8373 055600 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8374 055601 000 .BYTE 0 ; /UNUSED
8375 055602 000000 .WORD 0 ; /BUFFER ADDRESS #1
8376 055604 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8377 055605 000 .BYTE 0 ; /UNUSED
8378 055606 000000 .WORD 0 ; /BUFFER ADDRESS #1
8379 055610 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8380 055611 000 .BYTE 0 ; /UNUSED
8381 055612 000000 .WORD 0 ; /BUFFER ADDRESS #1
8382 055614 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8383 055615 000 .BYTE 0 ; /UNUSED
8384 055616 000000 .WORD 0 ; /BUFFER ADDRESS #1
8385 055620 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8386 055621 000 .BYTE 0 ; /UNUSED
8387 055622 000000 .WORD 0 ; /BUFFER ADDRESS #1
8388 055624 000 .BYTE 0 ; /EXTENDED ADDRESS BITS
8389 055625 000 .BYTE 0 ; /UNUSED
8390

```

M16

DMDT MACY11 27(654) 13-DEC-77 12:58 PAGE 177
 DRLPA.P11 DMDT -- DEDICATED MODE DISPATCH TABLE

SEQ 0207

8391	055626	055656	.WORD	JOB4R	;/RCLR (ADDR OF LIST OF RANDOM CHAN)
8392	055630	000	.BYTE	0	;/RCL EXTENDED ADR. BITS
8393	055631	000	.BYTE	0	;/UNUSED.
8394					
8395	055632	000	JOB4S: .BYTE	0	;/CHANNEL START ADDRESS
8396	055633	000	.BYTE	0	;/NUMBER OF CHANNELS
8397					
8398					
8399	055634	000	.BYTE	0	;/CHANNEL INCREMENT
8400	055635	000	.BYTE	0	;/FATAL ERROR MASK
8401					
8402	055636	000000	.WORD	0	;/DELAY
8403	055640	000000	.WORD	0	;/SAMPLE RATE
8404					
8405	055642	000	.BYTE	0	;/STWD
8406	055643	000	.BYTE	0	;/EMWD EVENT MARK DIGITAL INT WD#
8407					
8408	055644	000000	.WORD	0	;/ST MSK
8409	055646	000000	.WORD	0	;/EM MSK EVENT MARK DIGITAL IPUT MASK.
8410					
8411					;/END REG. TABLE
8412	055650	000003	.WORD	03	;/CONTAINS STOP CODE FOR THIS JOB.
8413	055652	000000	.WORD	0	;/CONTAINS LOOP COUNT FOR JOB.(NO OF TIMES THUR BUFFER.
8414					
8415	055654	000000	JOB4U: .WORD	0	;/USW USER STATUS WORD
8416					
8417					
8418	055656	000007	JOB4R: .BLKW	7	
8419					
8420					
8421			;/**		
8422			;/**		REQUEST DESCRIPTOR ARRAY FOR JOB #5
8423			;/**		
8424			;/**		THIS TABLE WILL BE ALERED BY THIS PROGRAM
8425			;/**		AND BY THE LPA-11 OPTION.
8426			;/**		
8427	055674	000000	JOB5: .WORD	0	;/MODE INFORMATION.
8428	055676	000000	.WORD	0	;/WORD COUNT
8429	055700	055772	.WORD	0+JOB5U	;/USW (USER STATUS WORD)
8430	055702	000	.BYTE	0	;/USW (EXTENDED ADDR. BITS)
8431	055703	000	.BYTE	0	;/VBM (VALID BUFFER MASK)
8432	055704	000000	.WORD	00	;/BUFFER ADDRESS #0
8433	055706	000	.BYTE	00	;/EXTENDED ADDRESS BITS
8434	055707	000	.BYTE	00	;/UNUSED
8435	055710	000000	.WORD	00	;/BUFFER ADDRESS #1
8436	055712	000	.BYTE	00	;/EXTENDED ADDRESS BITS
8437	055713	000	.BYTE	00	;/UNUSED
8438	055714	000000	.WORD	00	;/BUFFER ADDRESS #1
8439	055716	000	.BYTE	00	;/EXTENDED ADDRESS BITS
8440	055717	000	.BYTE	00	;/UNUSED
8441	055720	000000	.WORD	00	;/BUFFER ADDRESS #1
8442	055722	000	.BYTE	00	;/EXTENDED ADDRESS BITS
8443	055723	000	.BYTE	00	;/UNUSED
8444	055724	000000	.WORD	00	;/BUFFER ADDRESS #1

B01

DMDT MACY11 27(654) 13-DEC-77 12:58 PAGE 178
 DRLPA.P11 DMDT -- DEDICATED MODE DISPATCH TABLE

SEQ 0208

0445	055726	000	.BYTE	0	;/EXTENDED ADDRESS BITS
0446	055727	000	.BYTE	0	;/UNUSED
0447	055730	000000	.WORD	0	;/BUFFER ADDRESS #i
0448	055732	000	.BYTE	0	;/EXTENDED ADDRESS BITS
0449	055733	000	.BYTE	0	;/UNUSED
0450	055734	000000	.WORD	0	;/BUFFER ADDRESS #1
0451	055736	000	.BYTE	0	;/EXTENDED ADDRESS BITS
0452	055737	000	.BYTE	0	;/UNUSED
0453	055740	000000	.WORD	0	;/BUFFER ADDRESS #1
0454	055742	000	.BYTE	0	;/EXTENDED ADDRESS BITS
0455	055743	000	.BYTE	0	;/UNUSED
0456					
0457	055744	055774	.WORD	JOB5R	;/RCLR (ADDR OF LIST OF RANDOM CHAN)
0458	055746	000	.BYTE	0	;/RCL EXTENDED ADR. BITS
0459	055747	000	.BYTE	0	;/UNUSED.
0460					
0461	055750	000	JOB5S: .BYTE	0	;/CHANNEL START ADDRESS
0462	055751	000	.BYTE	0	;/NUMBER OF CHANNELS
0463					
0464					
0465	055752	000	.BYTE	0	;/CHANNEL INCREMENT
0466	055753	000	.BYTE	0	;/FATAL ERROR MASK
0467					
0468	055754	000000	.WORD	0	;/DELAY
0469	055756	000000	.WORD	0	;/SAMPLE RATE
0470					
0471	055760	000	.BYTE	0	;/STWD
0472	055761	000	.BYTE	0	;/EMWD EVENT MARK DIGITAL INT WDM
0473					
0474	055762	000000	.WORD	0	;/ST MSK
0475	055764	000000	.WORD	0	;/EM MSK EVENT MARK DIGITAL INPUT MASK.
0476					
0477					;/END REG. TABLE
0478	055766	000003	.WORD	03	;/CONTAINS STOP CODE FOR THIS JOB.
0479	055770	000000	.WORD	0	;/CONTAINS LOOP COUNT FOR JOB. (NO OF TIMES THUR BUFFER.
0480					
0481	055772	000000	JOB5U: .WORD	0	;/USW USER STATUS WORD
0482					
0483					
0484	055774	000007	JOB5R: .BLKW	7	
0485					
0486					
0487					;/** REQUEST DESCRIPTOR ARRAY FOR JOB #6
0488					;/**
0489					;/** THIS TABLE WILL BE ALERED BY THIS PROGRAM
0490					;/** AND BY THE LPA-11 OPTION.
0491					;/**
0492					
0493	056012	000000	JOB6: .WORD	0	;/MODE INFORMATION.
0494	056014	000000	.WORD	0	;/WORD COUNT
0495	056016	056110	.WORD	0+JOB6U	;/USW (USER STATUS WORD)
0496	056020	000	.BYTE	0	;/USW (EXTENDED ADDR. BITS)
0497	056021	000	.BYTE	0	;/VBM (VALID BUFFER MASK)
0498	056022	000000	.WORD	0	;/BUFFER ADDRESS #0

8499	056024	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8500	056025	000	.BYTE	0	;/UNUSED
8501	056026	000000	.WORD	0	;/BUFFER ADDRESS #1
8502	056030	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8503	056031	000	.BYTE	0	;/UNUSED
8504	056032	000000	.WORD	0	;/BUFFER ADDRESS #1
8505	056034	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8506	056035	000	.BYTE	0	;/UNUSED
8507	056036	000000	.WORD	0	;/BUFFER ADDRESS #1
8508	056040	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8509	056041	000	.BYTE	0	;/UNUSED
8510	056042	000000	.WORD	0	;/BUFFER ADDRESS #1
8511	056044	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8512	056045	000	.BYTE	0	;/UNUSED
8513	056046	000000	.WORD	0	;/BUFFER ADDRESS #1
8514	056050	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8515	056051	000	.BYTE	0	;/UNUSED
8516	056052	000000	.WORD	0	;/BUFFER ADDRESS #1
8517	056054	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8518	056055	000	.BYTE	0	;/UNUSED
8519	056056	000000	.WORD	0	;/BUFFER ADDRESS #1
8520	056060	000	.BYTE	0	;/EXTENDED ADDRESS BITS
8521	056061	000	.BYTE	0	;/UNUSED
8522					
8523	056062	056112	.WORD	JOB6R	;/RCLR (ADDR OF LIST OF RANDOM CHAN)
8524	056064	000	.BYTE	0	;/RCL EXTENDED ADR. BITS
8525	056065	000	.BYTE	0	;/UNUSED.
8526					
8527	056066	000	.BYTE	0	;/CHANNEL START ADDRESS
8528	056067	000	.BYTE	0	;/NUMBER OF CHANNELS
8529					
8530					
8531	056070	000	.BYTE	0	;/CHANNEL INCREMENT
8532	056071	000	.BYTE	0	;/FATAL ERROR MASK
8533					
8534	056072	000000	.WORD	0	;/DELAY
8535	056074	000000	.WORD	0	;/SAMPLE RATE
8536					
8537	056076	000	.BYTE	0	;/STWD
8538	056077	000	.BYTE	0	;/EMWD EVENT MARK DIGITAL INT WD#
8539					
8540	056100	000000	.WORD	0	;/ST MSK
8541	056102	000000	.WORD	0	;/EM MSK EVENT MARK DIGITAL INPUT MASK.
8542					
8543					
8544	056104	000003	.WORD	03	;/CONTAINS STOP CODE FOR THIS JOB.
8545	056106	000000	.WORD	0	;/CONTAINS LOOP COUNT FOR JOB.(NO OF TIMES THUR BUFFER.
8546					
8547	056110	000000	.WORD	0	;/USW USER STATUS WORD
8548					
8549					
8550	056112	000007	.BLKW	7	
8551					
8552					

;**

```

0553      : ** REQUEST DESCRIPTOR ARRAY FOR JOB #7
0554      : ** THIS TABLE WILL BE ALERED BY THIS PROGRAM
0555      : ** AND BY THE LPA-11 OPTION.
0556      : **
0557      : **
0558      : **
0559      056130 000000 JOB7: .WORD 0 ;/MODE INFORMATION.
0560      056132 000000 .WORD 0 ;/WORD COUNT
0561      056134 056226 .WORD 0+JOB7U ;/USW (USER STATUS WORD)
0562      056136 000 .BYTE 0 ;/USW (EXTENDED ADDR. BITS)
0563      056137 000 .BYTE 0 ;/VBM (VALID BUFFER MASK)
0564      056140 000000 .WORD 0 ;/BUFFER ADDRESS #0
0565      056142 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0566      056143 000 .BYTE 0 ;/UNUSED
0567      056144 000000 .WORD 0 ;/BUFFER ADDRESS #1
0568      056146 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0569      056147 000 .BYTE 0 ;/UNUSED
0570      056150 000000 .WORD 0 ;/BUFFER ADDRESS #1
0571      056152 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0572      056153 000 .BYTE 0 ;/UNUSED
0573      056154 000000 .WORD 0 ;/BUFFER ADDRESS #1
0574      056156 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0575      056157 000 .BYTE 0 ;/UNUSED
0576      056160 000000 .WORD 0 ;/BUFFER ADDRESS #1
0577      056162 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0578      056163 000 .BYTE 0 ;/UNUSED
0579      056164 000000 .WORD 0 ;/BUFFER ADDRESS #1
0580      056166 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0581      056167 000 .BYTE 0 ;/UNUSED
0582      056170 000000 .WORD 0 ;/BUFFER ADDRESS #1
0583      056172 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0584      056173 000 .BYTE 0 ;/UNUSED
0585      056174 000000 .WORD 0 ;/BUFFER ADDRESS #1
0586      056176 000 .BYTE 0 ;/EXTENDED ADDRESS BITS
0587      056177 000 .BYTE 0 ;/UNUSED
0588
0589      056200 056230 .WORD JOB7R ;/RCLR (ADDR OF LIST OF RANDOM CHAN)
0590      056202 000 .BYTE 0 ;/RCL EXTENDED ADR. BITS
0591      056203 000 .BYTE 0 ;/UNUSED.
0592
0593      056204 000 JOB7S: .BYTE 0 ;/CHANNEL START ADDRESS
0594      056205 000 .BYTE 0 ;/NUMBER OF CHANNELS
0595
0596
0597      056206 000 .BYTE 0 ;/CHANNEL INCREMENT
0598      056207 000 .BYTE 0 ;/FATAL ERROR MASK
0599
0600      056210 000000 .WORD 0 ;/DELAY
0601      056212 000000 .WORD 0 ;/SAMPLE RATE
0602
0603      056214 000 .BYTE 0 ;/STWD
0604      056215 000 .BYTE 0 ;/EMWD EVENT MARK DIGITAL INT WD#
0605
0606      056216 000000 .WORD 0 ;/ST MSK

```

8607	056220	000000	.WORD	0	;/EM MSK EVENT MARK DIGITAL IPUT MASK.
8608					
8609			;/END REG. TABLE		
8610	056222	000003	.WORD	03	;/CONTAINS STOP CODE FOR THIS JOB.
8611	056224	000000	.WORD	0	;/CONTAINS LOOP COUNT FOR JOB.(NO OF TIMES THUR BUFFER.
8612					
8613	056226	000000	J0B7U: .WORD	0	;/USW USER STATUS WORD
8614					
8615					
8616	056230	000007	J0B7R: .BLKW	7	
8617					
8618	056246		FRECOR: :		
8619	056246		BUFFER:		
8620	056246	000401	BUFF0: .BLKW	257.	
8621	057250	000401	BU:F1: .BLKW	257.	
8622	060252	000401	BU:F2: .BLKW	257.	
8623	061254	000401	BU:F3: .BLKW	257.	
8624	062256	000200	BU:F4: .BLKW	128.	
8625	062656	000200	BU:F5: .BLKW	128.	
8626					
8627					
8628					
8629					
8630					
8631					
8632					
8633					
8634					
8635					
8636					
8637					
8638					
8639					
8640	000001				

W A R N I N G !!!!!!!!!!!!!!!

THE FREE CORE AREA BETWEEN HERE AND PROGRAM
 "DRLPX0" MAY BE USED BY CERTAIN PARTS OF THIS
 PROGRAM. YOU MAY NOT PLACE ANY CODE HERE.
 LAST AVAILIBLE SPACE TO PUT CODE IS BEFORE
 LOCATION "FRECOR" OR AFTER ADDRESS "65500".

IF YOU HAVE TO VIOLATE THIS,DON'T ALLOW ROUTINE
 "MAKEI" TO BE RUN!

.END

H01

BIT13 = 020000	147#	1779	1941	2110	2281	2427	2573	2719	2865	3011	3157	3303	3449	3595	3741	3887	4033
BIT14 = 040000	146#	972	1437	2089	2281	2427	2573	2719	2865	3011	3157	3303	3449	3595	3741	3887	4033
	4146	4148	4265	4452	5432	5532	6051	6690	6717	6972	7427	7882	8337	8792	9247	9702	10157
BIT15 = 100000	145#	1443	1772	1785	1798	1845	1934	1947	1960	2007	3617	3630	3643	3656	3669	3682	3695
	3690	3858	3911	4270	4463	5432	5514	6672	6676	6690	6717	6980	7440	7895	8350	8805	9260
BIT2 = 000004	168#	2110	2133	2249	2303	3337	3637	3787	4459	4757	4823	5514	5540	5570	5600	5630	5660
BIT3 = 000010	167#	977	2089	2145	2960	3317	3766	4200	4202	4443	4701	4762	4780	4810	4840	4870	4900
	4875	4893	4988	5006	5101	5119	5214	5232	5327	5406	5532	5540	5570	5600	5630	5660	5690
BIT4 = 000020	166#	1792	1911	1954	2133	3225	3317	3337	3354	3370	3382	4310	4437	4564	4691	4818	4945
	4763	4781	4876	4894	4989	5007	5102	5120	5215	5233	5347	5435	5551	5667	5783	5899	6015
	5669	7241															
BIT5 = 000040	165#	1448	1580	2145	2177	2395	2449	4470	4870	4936	5570	5686	6984				
	7197	7214															
BIT6 = 000100	164#	1779	1941	2089	2133	2281	2427	2573	2719	2865	3212	3317	3624				
	3766	3889	4310	4650	4703	6834	6835	6861									
BIT7 = 000200	163#	977	1772	1785	1798	1810	1823	1845	1859	1934	1947	1960	1972				
	1985	2007	2021	2177	2207	2313	2459	2541	2595	2605	2751	2897	2998				
	3010	3022	3034	3205	3218	3231	3243	3256	3370	3429	3441	3453	3465				
	3617	3630	3643	3655	3668	3690	3703	3819	3921	4021	4033	4045	4057				
	4274	4279	4284	4310	4475	4500	4709	4781	4894	4983	5007	5049	5120				
BIT8 = 000400	5233	5337	5416	5502	5540	5558	5590	6724	6916								
	162#	1792	1954	2110	2687	2741	3225	3337	3637	3787	4480	4703	4763				
	4781	4876	4894	4989	5007	5096	5102	5120	5162	5215	5233	5498	5610				
	5686	6701															
BIT9 = 001000	161#	1631	1779	1941	2089	2833	2887	3212	3624	3766	4261	4485	5209				
	5275	5324	5540	5630	6701												
BPTVEC = 000014	177#																
BUFFER 056246	6504	7598	8619#														
BUFF0 056246	4768	4807	4827	4881	4920	4940	4994	5033	5053	5107	5146	5166	5220				
	5259	5279	5331	5410	5503	5559	8620#										
BUFF1 057250	4769	4882	4995	5108	5221	5332	5411	8621#									
BUFF2 060252	4786	4788	4801	4826	4899	4901	4914	4939	5012	5014	5027	5052	5125				
	5127	5140	5165	5238	5240	5253	5278	5333	5412	5522	5578	5598	5618				
	5638	8622#															
BUFF3 061254	4787	4900	5013	5126	5239	5545	8623#										
BUFF4 062256	8624#																
BUFF5 062656	8625#																
CAINC = 000000	7949#	7972	7976	7984	7988	7999	8002	8016	8019	8036	8052						
CHAN 034466	5711#	5720*	6503#	6507	6538*	6539	6569#	7756									
CHANF 034476	5358#	5446*	5707*	5716*	6539	6573#	6760*										
CHANFR 034502	6575#																
CHANNO 034504	6576#																
CHANS 034474	5357*	5445*	5706*	5711	5715*	5720	6503	6572#	6759*								
CHANSR 034500	6574#																
CHANU 034454	5344*	5423*	5698*	6505	6564#	6755*	6758										
CHAN1 034464	6507*	6508*	6510	6537*	6568#												
CHAN7 034456	6565#																
CKSWR = 104407	5950	5976	6049	6050	7583#												
CLKINT 040616	889	890	7420#														
CLKTMP 002006	697	698#															
CLR = 000000	7953#	7967	7972	7979	7984	7991	7994	7999	8002	8008	8011	8016	8019				
CR = 000015	85#	6310	6320														
CRLF = 000200	86#	909	6281	6320													
DACO 001626	601#	823*	827*	3043	3045	3087	3089	3132	3134	4211	4227						

DAC1	001630	602#	824*	828*	3054	3056	3098	3100	3143	3145	4219	4234		
DAC2	001632	603#	825*	829*	3065	3067	3109	3111	3154	3156				
DAC3	001634	604#	826*	830*	3076	3078	3120	3122	3165	3167				
DDISP =	177570	92#	250	731										
DEVLST	054130	4354	4390	4400	4493*	4496	4569	4570*	4571*	4597	4631	4632*	4633*	4648
		4682	4683*	4684*	6799	6800*	6801*	7914#						
DFD	042606	337	344	351	358	365	372	379	386	393	400	414	421	428
		435	442	448	7762#									
DH1	042240	335	7709#											
DH17	042321	433	7721#											
DH2	042261	342	363	370	377	384	391	398	412	419	426	7713#		
DH20	042361	440	7728#											
DH21	042427	446	7735#											
DH3	042312	349	356	7719#										
DISPLA	001142	250#	731*	739*	5953*	6095*								
DISPRE	000174	62#	739											
DMA:T	050124	4259	7435	7914#										
DMDSIZ=	000022	7947#	7967	7972	7979	7984	7991	7999	8008	8016	8028	8036	8044	8052
DMDT	054156	7916#	7965#											
DONEC	055004	6810	8079#											
DON&Q	054664	8078#												
DRCR1	001636	608#	787*	2257	2270	2284	2286	2295	2297	2313	2334	2371		
DRCR2	001644	613#	794*	2403	2416	2430	2432	2441	2443	2459	2480	2517		
DRCR3	001652	618#	801*	2549	2562	2576	2578	2587	2589	2605	2626	2663		
DRCR4	001660	623#	808*	2695	2708	2722	2724	2733	2735	2751	2772	2809		
DRCR5	001666	628#	815*	2841	2854	2868	2870	2879	2881	2897	2918	2955		
DRIA1	001642	610#	789*	791*	2281	2325	2343	2348	2356	2362				
DRIA2	001650	615#	796*	798*	2427	2471	2489	2494	2502	2508				
DRIA3	001656	620#	803*	805*	2573	2617	2635	2640	2648	2654				
DRIA4	001664	625#	810*	812*	2719	2763	2781	2786	2794	2800				
DRIA5	001672	630#	817*	819*	2865	2909	2927	2932	2940	2946				
DRLPX0=	*****	1434#	1435	6729										
DRLPX1=	*****	970#	971											
DRLPX2=	*****	14#	6970											
DROA1	001640	609#	788*	790*	2279	2311	2346	2360						
DROA2	001646	614#	795*	797*	2425	2457	2492	2506						
DROA3	001654	619#	802*	804*	2571	2603	2638	2652						
DROA4	001662	624#	809*	811*	2717	2749	2784	2798						
DROA5	001670	629#	816*	818*	2863	2895	2930	2944						
DR11K1	001572	574#	786	4461										
DR11K2	001600	577#	793	4472										
DR11K3	001602	578#	800	4477										
DR11K4	001604	579#	807	4482										
DR11K5	001606	580#	814	4487										
OSWR =	177570	91#	249	730										
DT1	042516	336	7747#											
DT17	042540	434	7753#											
DT2	042524	343	364	371	378	385	392	399	413	420	427	7749#		
DT20	042552	441	7756#											
DT21	042566	447	7759#											
DT3	042534	350	357	7752#										
D.O.CQ	054247	7983#	7989											
D.OCS	054224	7978#	7982											
D.O&Q	054201	7971#	7977											

G G

G G G

NO1

RTEMP1 034526
RO =%000000

6521*	6522	6527*	6528*	6529	6531*	6532*	6533	6585*	773	774	775	776
95*	748*	750	751*	766*	767	768	771*	772	800*	801	802	803
777	786*	787	788	789	793*	794	795	796	822	823	824	825
807*	808	809	810	814*	815	816	817	821*	842	843	844	845
826	832*	833	834	837*	838	839	840	841	863	864	973*	974*
854*	855	856	857	858	859	860	861	862	1387*	1388*	1400*	1402*
1130*	1133*	1178*	1181*	1249*	1253*	1320*	1324*	1387*	4354*	4355*	4356	4359*
1440*	1444*	1445*	1579*	1582*	1656*	1659*	4313*	4316	4751*	4752*	4753	4761*
4363*	4389*	4398*	4498*	4502*	4538*	4542*	4599*	4628*	4874*	4912*	4915	4916*
4799*	4802	4803*	4807*	4825*	4828*	4864*	4865*	4866	5029*	5033*	5051*	5054*
4920*	4938*	4941*	4977*	4978*	4979	4987*	5025*	5028	5167*	5203*	5204*	5205
5090*	5091*	5092	5100*	5138*	5141	5142*	5146*	5164*	5322*	5351*	5396*	5398
5213*	5251*	5254	5255*	5259*	5277*	5280*	5317*	5319	5553*	5572*	5592*	5598*
5401*	5424*	5439*	5487*	5489	5491	5496*	5516*	5539*	5925*	5996	5997*	5998*
5632*	5685*	5777*	5780	5881	5891*	5895	5911	5912	6022*	6026	6262	6263*
6005*	6006*	6007*	6008*	6009*	6010	6015	6020*	6022*	6378	6379	6381*	6382*
6268	6273	6276*	6332	6336*	6339	6355*	6366	6374*	6748*	6754*	6768*	6810
6383	6405*	6421	6446*	6597*	6598	6603*	6668*	6748*	6876*	6877*	6878	6879
6812*	6817	6818*	6824*	6829*	6853	6854*	6864*	6867	7146*	7147*	7149	7162*
6880*	6923*	7098	7099*	7110*	7113	7121*	7145	7146*	7291*	7314	7317*	7428*
7177	7178*	7179*	7181	7184	7219*	7272	7275*	7291*	7557*	7558*	7598*	7618
7430*	7441*	7443*	7552	7553*	7554	7555*	7556*	7557*	4779*	4800*	4804*	4808*
7626*	749*	752	4266*	4268*	4272*	4277*	4390*	4400*	4939*	4943	4945	5005*
96*	4826*	4830	4832	4892*	4913*	4917*	4921*	4923*	5143*	5147*	5149*	5165*
4810*	5030*	5034*	5036*	5052*	5056	5058	5118*	5139*	5284	5710*	5719*	5882
5026*	5171	5231*	5252*	5256*	5260*	5262*	5278*	5282	6348*	6351	6354*	6367
5169	5896	5900	5924*	6333	6337*	6341*	6343*	6345*	6662*	6705	6736*	6753*
5895*	6422	6445*	6504*	6510*	6524*	6625*	6646	6660*	6981*	7029*	7030	7035
6404*	6813*	6868	6881*	6903*	6922*	6976	6977*	6978*	7276*	7281*	7284	7315
6767*	7064*	7233	7234*	7237*	7243*	7247*	7273	7276*	5027*	5053*	5056	5059
7037	7324*	7327	7334*	7605*	7607*	7608*	7626	7626	5356*	5362*	5444*	5450*
7318*	4801*	4827*	4830	4833	4914*	4940*	4943	4946	6334	6338*	6342*	6346*
97*	5166*	5169	5172	5253*	5279*	5282	5285	5356*	6654	6656*	6728	6758*
5140*	5898*	5901	5908*	5909*	5910	5915*	5923*	5923*	6654	6656*	6728	6758*
5894*	6353*	6423	6444*	6626*	6627	6628*	6649	6649	6912	6913	6921*	6921*
6352	6821	6823*	6869	6905*	6906*	6907*	6911*	6911*	5516	5539	5553	5572
6814*	4392*	5352*	5355*	5440*	5443*	5488*	5489*	5516	5843*	5848*	5849*	5850
98*	5632	5705*	5714*	5824	5833*	5839*	5840*	5843*	6208	6209*	6210	6213*
5612	5892*	5893*	5907*	5910*	5919*	5920*	5922*	5922*	6664	6704*	6705*	6706
5884	6220	6222*	6224*	6424	6443*	6646*	6662	6662	6823	6870	6879*	6904*
6218	6736	6756*	6759	6760	6764*	6815*	6816*	6816*	5847*	5855*	5858*	6425
6735*	99*	5825	5827*	5828*	5829*	5830	5831*	5845	7097	7122*	7127*	6442*
R4 =%000004	6647*	6648*	6656	6871	6908*	6909*	6913	6918*	5855*	5858*	6425	6442*
R5 =%000005	100*	969*	1433*	1757*	1772*	1783*	1785*	1796*	1808*	1810*	1820*	1823*
	1827*	1842*	1845*	1857*	1859*	1867*	1919*	1934*	1947*	1958*	1960*	1970*
	1972*	1982*	1985*	1989*	2004*	2007*	2019*	2021*	2029*	2093*	2103*	2115*
	2125*	2127*	2136*	2138*	2149*	2151*	2160*	2162*	2172*	2174*	2176*	2187*
	2199*	2201*	2207*	2217*	2219*	2257*	2270*	2279*	2281*	2284*	2286*	2295*
	2311*	2313*	2325*	2334*	2343*	2346*	2348*	2356*	2360*	2362*	2371*	2403*
	2425*	2427*	2430*	2432*	2441*	2443*	2457*	2459*	2471*	2480*	2489*	2492*
	2502*	2506*	2508*	2517*	2549*	2562*	2571*	2573*	2576*	2578*	2587*	2589*
	2605*	2617*	2626*	2635*	2638*	2640*	2648*	2652*	2654*	2663*	2695*	2708*
	2719*	2722*	2724*	2733*	2735*	2749*	2751*	2763*	2772*	2781*	2784*	2786*

R1 =%000001

R2 =%000002

R3 =%000003

R4 =%000004

R5 =%000005

SR2	001564	560#	2303	2449	2595	2741	2887	3549	3911	4148	4202	4823	4936	5049
		5162	5275	5347	5435	5669								
STACK =	001100	78#	711	886	4666	5679								
START	002016	68	704#											
STKLMT=	177774	89#												
STPCOD=	000073	690#												
STREG1	001614	591#	767*	1757	1772	1763	1785	1796	1798	1808	1810	1823	1827	1842
		1845	1857	1859										
STREG2	001620	595#	833*	1919	1934	1945	1947	1958	1960	1970	1972	1985	1989	2004
		2007	2019	2021										
SWR	001140	249#	709	730*	732	738*	745*	901	4251	4261	5954	5961	5973	5977
		6051	6065	6067	6073	6080	6109	6146*	6427	6440*				
SWREG	000176	63#	738	901	6109	6122								
SW0	= 000001	142#												
SW00	= 000001	132#	142											
SW01	= 000002	131#	141											
SW02	= 000004	130#	140											
SW03	= 000010	129#	139											
SW04	= 000020	128#	138											
SW05	= 000040	127#	137											
SW06	= 000100	126#	136											
SW07	= 000200	125#	135											
SW08	= 000400	124#	134											
SW09	= 001000	123#	133											
SW1	= 000002	141#												
SW10	= 002000	122#												
SW11	= 004000	121#												
SW12	= 010000	120#												
SW13	= 020000	119#												
SW14	= 040000	118#												
SW15	= 100000	117#												
SW2	= 000004	140#												
SW3	= 000010	139#												
SW4	= 000020	138#												
SW5	= 000040	137#												
SW6	= 000100	136#												
SW7	= 000200	135#												
SW8	= 000400	134#												
SW9	= 001000	133#												
SYNC	034112	6481#												
TABLE	054642	6900*	6903	8067*										
TAXING	001774	684#	6836*	6837*	6838*	6843*	6872*							
TBITVE=	000014	175#												
TIME	040614	7406#	7409	7414*	7418#	7420*								
TKVEC =	000060	182#												
TODOC	054640	6817*	6851	6859*	8059#									
TODOQ	054520	6807	6813	6854	8058#									
TQLER	034512	5360*	5448*	5709*	5718*	6528	6531	6532	6535	6579#				
TPVEC =	000064	183#												
TRAPVE=	000034	181#	717*	718*										
TRTVEC=	000014	176#												
TST1	003504	918#												
TST10	007254	2143	2155	2166	2193	2220	2248#							
TST11	007762	2305	2320	2329	2338	2352	2366	2394#						

F02

\$OBLK	031454	5892	5926	5934*											
\$DEVCT	001204	275*													
\$DEVm	001252	311*													
\$DOAGN	031002	5745	5778	5784*											
\$OTBL	031444	5895	5930*												
\$ENDAD	030772	194	5780*												
\$ENDCT	030634	5747*													
\$ENULL	031006	5787*													
\$ENV	001214	280*	899	5966	6264	6370	6394								
\$ENVM	001215	281*	743	6266	6271	6372									
\$EOP	030600	5737*													
\$EOPCT	030626	5744*	5748												
\$ERFLG	001103	232*	5951*	5987	6039	6069	6071	6077*	6099						
\$ERMAX	001115	238*	723*	6071	6094*	6099									
\$ERROR	031464	715	5949*												
\$ERRPC	001116	239*	5958*	5959*	5960	5987	6001	7747	7749	7752	7753	7759			
\$ERRTB	001256	329*	6009												
\$ERRTY	031660	5963	5994*												
\$ERTTL	001112	236*	5957*	5987											
\$ESCAP	001162	260*	722*	5980	5982	5987	6093*								
\$ETABL	001214	279*													
\$ETEND	001256	221	313*												
\$FATAL	001176	272*	6398*												
\$FFLG	033702	6361*	6364*	6392	6401*	6409*									
\$FILLC	001156	257*	6289	6320											
\$FILLS	001155	256*	6320												
\$GDADR	001120	240*													
\$GDDAT	001124	242*	1000*	1002	1007*	1009	1015	1021	1055*	1246*	1258	1316*	1330	1472*	
		1481	1563*	1567	1595*	1597	1635*	1640	1671*	1673	1773*	1779*	1783	1786	
		1792*	1796	1799	1805*	1808	1820*	1823*	1828	1835*	1842	1845*	1847	1854*	
		1857	1935*	1941*	1945	1948	1954*	1958	1961	1967*	1970	1962*	1985*	1990	
		1997*	2004	2007*	2009	2016*	2019	2089*	2093	2104	2110*	2113	2116	2122*	
		2125	2133*	2136	2139	2145*	2149	2151	2157*	2160	2168*	2174	2177*	2179	
		2184*	2187	2195*	2201	2207*	2208	2214*	2217	2254*	2257	2279	2281*	2284	
		2286	2292*	2295	2308*	2311	2313*	2314	2322*	2325	2343*	2346	2348	2356	
		2357*	2360	2371	2400*	2403	2425*	2427*	2430	2432	2438*	2441	2454*	2457	
		2459*	2460	2468*	2471	2489*	2492	2494	2502	2503*	2506	2517	2546*	2549	
		2571	2573*	2576	2578	2584*	2587	2600*	2603	2605*	2606	2614*	2617	2635*	
		2638	2640	2648	2649*	2652	2663	2692*	2695	2717	2719*	2722	2724	2730*	
		2733	2746*	2749	2751*	2752	2760*	2763	2781*	2784	2786	2794	2795*	2798	
		2809	2838*	2841	2863	2865*	2868	2870	2876*	2879	2892*	2895	2897*	2898	
		2906*	2909	2927*	2930	2932	2940	2941*	2944	2955	2984*	2987	3004*	3007	
		3011	3017*	3020	3023	3029*	3032	3040*	3043	3045	3054	3056	3065	3067	
		3076	3078	3084*	3087	3089	3098	3100	3109	3111	3120	3122	3129*	3132	
		3134	3143	3145	3154	3156	3165	3167	3187*	3206*	3212*	3216	3219	3225*	
		3229	3232	3238*	3241	3253*	3256*	3261	3317*	3321	3331	3337*	3340	3343	
		3349*	3352	3354*	3355	3361*	3367	3370*	3372	3377*	3380	3382*	3383	3415*	
		3418	3435*	3438	3442	3448*	3451	3454	3460*	3463	3471*	3474	3476	3485	
		3487	3493*	3496	3498	3506	3508	3514*	3517	3519	3528	3530	3552*	3558	
		3560*	3561	3574	3576*	3618*	3624*	3628	3631	3637*	3641	3644	3650*	3653	
		3665*	3668*	3673	3680*	3687	3690*	3692	3698*	3701	3766*	3770	3781	3787*	
		3790	3793	3799*	3802	3810*	3816	3819*	3821	3826*	3829	3862*	3865	3886	
		3888	3889*	3892	3894	3900*	3903	3916*	3919	3921*	3922	3930*	3933	3951*	
		3954	3956	3965*	3968	3979	4007*	4010	4027*	4030	4034	4040*	4043	4046	

DRLPA.P11 CROSS REFERENCE TABLE

\$MTYP4	001241	306#												
\$MXCNT	032274	6088#	6098#											
\$NULL	001154	255#	6291	6320										
\$NWIST=	000001	915#	954#	956	1413#	1415	1495#	1497	1707#	1709	1869#	1871	2031#	2033
		2228#	2230	2374#	2376	2520#	2522#	2666#	2668	2812#	2814	2959#	2961#	3175#
		3269#	3271	3390#	3392	3537#	3539	3587#	3713#	3715	3837#	3839	3982#	3984
		4135#	4137	4188#	4190	4240#	4242	4298#	4300	4345#	4347	4380#	4382#	4524#
		4526	4558#	4560	4616#	4618	4671#	4673	4726#	4728	4839#	4841	4952#	4954
		5065#	5067	5178#	5180	5292#	5294	5367#	5369	5455#	5457	5652#	5654	5726#
\$OCNT	031234	5823#	5852#	5865#										
\$OFS =	040412	7330#	7530	7531	7537									
\$OMODE	031236	5818#	5822#	5827	5830#	5841#	5867#							
\$OUTLP	040252	1757	1783	1796	1808	1823	1842	1857	1919	1945	1958	1970	1985	2004
		2019	2093	2113	2125	2136	2149	2160	2172	2174	2187	2199	2201	2217
		2257	2284	2295	2311	2343	2346	2356	2360	2371	2403	2430	2441	2457
		2489	2492	2502	2506	2517	2549	2576	2587	2603	2635	2638	2648	2652
		2663	2695	2722	2733	2749	2781	2784	2794	2798	2809	2841	2868	2879
		2895	2927	2930	2940	2944	2955	2987	3007	3020	3032	3043	3054	3065
		3076	3087	3098	3109	3120	3132	3143	3154	3165	3190	3216	3229	3241
		3256	3321	3340	3352	3365	3367	3380	3418	3438	3451	3463	3474	3485
		3496	3506	3517	3528	3558	3560	3566	3568	3574	3576	3581	3583	3601
		3628	3641	3653	3668	3687	3701	3770	3790	3802	3814	3816	3829	3865
		3892	3903	3919	3951	3954	3965	3968	3979	4010	4030	4043	4055	4066
		4077	4089	4100	4113	4124	4157	4159	4165	4167	4173	4175	4180	4182
		4211	4213	4219	4221	4227	4229	4234	4236	7272#	7492			
\$OVER	032260	6052	6068	6076	6086	6095#								
\$PASS	001202	274#	742#	4249	5741#	5742#	5768	5787	6082	6099	6644#	6658#	6703#	6715#
\$PASTM	001006	219#												
\$POWER	034052	6450	6455#											
\$PUTS	041132	7526#												
\$PWDRN	033704	719	6419#	6447										
\$PWRMG	034040	6450#												
\$PWRUP	033756	6429	6435#											
\$QUES	001170	262#	5987	6167	6216	6232	6320							
\$RDCHR	032560	6180#	7584											
\$RDDEC=	***** U	7587												
\$RDLIN	032700	6208#	7585											
\$RDOCT	033334	6330#	7586											
\$ROSZ =	000010	6201#												
\$RESET	040504	4248	7374#	7384										
\$RTNAD	031004	5786#												
\$R2A =	***** U	7587												
\$SAD	040500	7352	7354#	7362#										
\$SAVRE=	***** U	7587												
\$SAVR6	034050	6428#	6436	6437#	6438#	6454#								
\$SCOPE	032014	713	6048#											
\$SETUP=	000117	703#	712	713	715	717	719	721	722	724	895	896	5739	5950
		5976	5984	6049	6104	6238								
\$STUP =	177777	703#												
\$SVLAD	032224	6060	6089#											
\$SVPC =	000244	192#	197											
\$SWR =	167400	31#	41	47	48	49	50	51	52	53	259	260	261	721
		722	724	725	919	967	1431	1541	1748	1910	2082	2249	2395	2541
		2687	2833	2979	3179	3310	3410	3546	3591	3759	3858	4002	4144	4197

K02

BAHE	8102#	8105	8108	8111	8114	8117	8120	8123	8168#	8171	8174	8177	8180	8183	8186
	8189	8234#	8237	8240	8243	8246	8249	8252	8255	8300#	8303	8306	8309	8312	8315
	8318	8321	8366#	8369	8372	8375	8378	8381	8384	8387	8432#	8435	8438	8441	8444
	8447	8450	8453	8498#	8501	8504	8507	8510	8513	8516	8519	8564#	8567	8570	8573
	8576	8579	8582	8585											
COMMEN	185#														
DAAT	4186#	4187													
DFC	331#	337	344	351	358	365	372	379	386	393	400	414	421	428	435
	442	448													
DRLT	4724#	4725	4838	4951	5064	5177									
ENDCOM	185#														
ENDPAS	5730#	5749													
ERROR	79#	942	1005	1012	1018	1024	1060	1065	1070	1075	1136	1142	1184	1190	1256
	1261	1327	1333	1391	1405	1458	1463	1476	1486	1570	1586	1600	1643	1663	1676
	1767	1777	1789	1802	1814	1831	1851	1862	1929	1939	1951	1964	1976	1993	2013
	2024	2097	2107	2119	2130	2142	2154	2165	2182	2192	2211	2222	2260	2273	2289
	2300	2317	2328	2337	2351	2365	2406	2419	2435	2446	2463	2474	2483	2497	2511
	2552	2565	2581	2592	2609	2620	2629	2643	2657	2698	2711	2727	2738	2755	2766
	2775	2789	2803	2844	2857	2873	2884	2901	2912	2921	2935	2949	2990	3001	3014
	3026	3037	3048	3059	3070	3081	3092	3103	3114	3125	3137	3148	3159	3170	3200
	3210	3222	3235	3247	3264	3325	3334	3346	3358	3375	3386	3421	3432	3445	3457
	3468	3479	3490	3501	3511	3522	3533	3611	3622	3634	3647	3659	3676	3696	3706
	3774	3784	3796	3807	3824	3834	3868	3881	3897	3908	3925	3936	3945	3959	3973
	4013	4024	4037	4049	4060	4071	4082	4094	4105	4118	4129	4281	4288	4296	4323
	4333	4343	4367	4376	4416	4507	4516	4520	4547	4555	4580	4594	4605	4614	4644
	4658	4692	4713	4722	4835	4948	5061	5174	5287	6467	6557	6847	6893	6987	6993
	7006	7047	7074	7188	7205										
ESCAPE	185#														
GETPRI	185#														
GETSWR	185#	896#													
INSTR2	5935#	5984													
LDGD	4186#	4327	4341	4365	4371	4545	4550	4578	4587	4603	4608	4641	4655	4711	4719
LPWT	4670#														
MOVEI	20#	1770	1783	1796	1808	1817	1825	1843	1857	1865	1932	1945	1958	1970	1979
	1987	2005	2019	2027	2101	2113	2125	2136	2149	2160	2174	2187	2205	2217	2267
	2276	2279	2284	2295	2311	2323	2331	2346	2360	2413	2422	2425	2430	2441	2457
	2469	2477	2492	2506	2559	2568	2571	2576	2587	2603	2615	2623	2638	2652	2705
	2714	2717	2722	2733	2749	2761	2769	2784	2798	2851	2860	2863	2868	2879	2895
	2907	2915	2930	2944	2996	3007	3020	3032	3043	3054	3065	3076	3087	3098	3109
	3120	3132	3143	3154	3165	3203	3216	3229	3241	3250	3258	3328	3340	3352	3367
	3380	3427	3438	3451	3463	3474	3485	3496	3506	3517	3528	3615	3628	3641	3653
	3662	3670	3688	3701	3709	3778	3790	3802	3816	3829	3875	3884	3886	3892	3903
	3919	3931	3939	3954	3968	4019	4030	4043	4055	4066	4077	4089	4100	4113	4124
	7375														
MOVEM	19#	1755	1781	1794	1806	1821	1840	1855	1917	1943	1956	1968	1983	2002	2017
	2091	2111	2123	2134	2147	2158	2170	2172	2185	2197	2199	2215	2255	2282	2293
	2309	2340	2344	2353	2358	2368	2401	2428	2439	2455	2486	2490	2499	2504	2514
	2547	2574	2585	2601	2632	2636	2645	2650	2660	2693	2720	2731	2747	2778	2782
	2791	2796	2806	2839	2866	2877	2893	2924	2928	2937	2942	2952	2985	3005	3018
	3030	3041	3051	3062	3073	3085	3095	3106	3117	3130	3140	3151	3162	3188	3214
	3227	3239	3254	3319	3338	3350	3363	3365	3378	3416	3436	3449	3461	3472	3482
	3494	3503	3515	3525	3555	3558	3563	3566	3571	3574	3578	3581	3599	3626	3639
	3651	3666	3685	3699	3768	3788	3800	3812	3814	3827	3863	3890	3901	3917	3948
	3952	3962	3966	3976	4008	4028	4041	4053	4064	4074	4087	4097	4111	4121	4154

MRDA	4157	4162	4165	4170	4173	4177	4180	4208	4211	4216	4219	4224	4227	4231	4234
MULT	7905#	8090	8156	8222	8288	8354	8420	8486	8552						
NEWST	185#	915	954	1413	1495	1707	1869	2031	2228	2374	2520	2666	2812	2959	3175
	3269	3390	3537	3587	3713	3837	3982	4135	4188	4240	4298	4345	4380	4524	4558
	4616	4671	4726	4839	4952	5065	5178	5292	5367	5455	5652	5726			
POP	185#	5921	6353	6404	6405	6440	6441								
PUSH	185#	5890	6332	6365	6367	6388	6421	6427							
REPORT	185#														
RWARN	4670#	4735	4848	4961	5074	5187	5301	5380							
R2D2	784#	785	792	799	806	813									
SCOPE	80#	966	1430	1540	1747	1909	2081	2248	2394	2540	2686	2832	2978	3178	3309
	3409	3545	3590	3758	3857	4001	4143	4196	4246	4307	4353	4387	4531	4565	4625
	4678	4743	4856	4969	5082	5195	5309	5388	5482	5665	5729				
SDRM	5510#	5511	5567	5587	5607	5627									
SETPRI	185#														
SETTRA	7567#	7576	7577	7578	7579	7581	7583	7584	7585	7586					
SETUP	185#	704													
SKIP	185#	1137	1143	1185	1328	1392	1406	1460	1466	1479	1484	1698	1778	1790	1803
	1815	1833	1940	1952	1965	1977	1995	2143	2155	2166	2193	2220	2305	2320	2329
	2338	2352	2366	2451	2466	2475	2484	2498	2512	2597	2612	2621	2630	2644	2658
	2743	2758	2767	2776	2790	2804	2889	2904	2913	2922	2936	2950	3082	3093	3104
	3115	3127	3138	3149	3160	3169	3211	3223	3236	3248	3262	3384	3502	3512	3523
	3548	3550	3623	3635	3648	3660	3678	3832	3898	3909	3913	3928	3937	3946	3960
	3974	4095	4106	4119	4127	4147	4149	4201	4203	4282	4289	4293	4324	4334	4339
	4368	4374	4378	4508	4510	4518	4548	4554	4581	4595	4606	4612	4645	4659	4691
	4715	4718	4824	4829	4937	4942	5050	5055	5163	5168	5276	5281	5348	5436	
SLASH	185#														
SPACE	185#														
STANDE	4569#	4631	4682	6799											
STARS	185#	190	202	204	211	224	265	268	915	917	954	965	1413	1429	1495
	1539	1707	1746	1869	1908	2031	2080	2228	2247	2374	2393	2520	2539	2666	2685
	2812	2831	2959	2977	3175	3177	3269	3308	3390	3408	3537	3544	3587	3589	3713
	3757	3837	3856	3982	4000	4135	4142	4188	4195	4240	4245	4298	4306	4345	4352
	4380	4386	4524	4530	4558	4564	4616	4624	4671	4677	4726	4742	4839	4855	4952
	4968	5065	5081	5178	5194	5292	5308	5367	5387	5455	5481	5652	5664	5726	5728
	5732	5793	5870	5937	5989	6036	6101	6104	6172	6201	6243	6322	6360	6417	6433
	7546														
SWRSU	185#	726#													
TORT	2226#	2227	2373	2519	2665	2811									
TOUT	489#	6990	7044	7071	7185	7202									
TRMTRP	7567#														
TSTE	8627#	8628													
TYPBIN	185#														
TYPDEC	185#	5768	5775												
TYPNAM	185#	891													
TYPNUM	185#														
TYPOCS	31#	185#													
TYPOCT	185#	6001	6025	6122	6630	6654	6664	7479	7618	7624					
TYPTXT	185#	5753	5758	5764	5771	6650	6708	6743	6774	6779	6961	7015	7019	7023	7250
	7460	7485	7590	7599	7609	7613	7620								
XY	953#	956	1412#	1415	1495#	1497	1706#	1709	1868#	1871	2030#	2033	3268#	3271	3712#
	3715														
ZK	1705#	1709	1867#	1871	2034	2230	2376	2522	2668	2814	2961	3272	3392	3716	3839

.SINLP	28#	7299
.SMMAC	18#	
.SOUTL	27#	7260
.SPOWE	31#	6415
.SRDOC	31#	6320
.SREAD	31#	6099
.SSB2D	31#	
.SSCOP	31#	6034
.STLKw	26#	7138
.STOUT	489#	7343
.STRAP	31#	7544
.STYPD	31#	5868
.STYPE	31#	6241
.STYPO	31#	5791

BLE	6540	6914													
BLOS	6211														
BLT	5854	5897	5913	6155	6196	6294	6530	6534							
BMI	2335	2481	2627	2773	2919	3943	4320	4362	4541	4577	4602	5904	7358		
BNE	710	733	753	894	898	902	975	1134	1140	1182	1254	1325	1441	1446	1453
	1471	1548	1559	1632	1751	1913	2085	2250	2336	2542	2688	2834	2981	3183	3313
	3412	3577	3584	3595	3762	3859	4004	4176	4183	4230	4237	4250	4252	4262	4269
	4275	4278	4399	4501	4503	4689	4700	4710	4754	4758	4805	4811	4867	4871	4918
	4924	4980	4984	5031	5037	5093	5097	5144	5150	5206	5210	5257	5263	5312	5320
	5336	5354	5363	5391	5399	5415	5442	5451	5492	5668	5752	5844	5902	5962	5967
	5999	6021	6052	6081	6110	6115	6136	6143	6150	6187	6193	6215	6221	6265	6272
	6274	6282	6290	6304	6311	6371	6377	6380	6397	6439	6465	6526	6548	6599	6608
	6671	6675	6702	6716	6766	6769	6811	6825	6844	6860	6889	6898	6979	6982	7003
	7005	7012	7014	7041	7057	7084	7108	7114	7198	7215	7238	7244	7378	7410	7415
	7431	7444	7597	7606											
BPL	1389	1403	1474	1583	1660	4364	4510	4543	4654	4718	5842	5888	5918	5974	6112
BR	6128	6183	6189	6259	6308	6883	7128	7405	7466						
	735	880	884	904	907	939	1137	1143	1185	1328	1392	1406	1460	1466	1479
	1614	1688	1698	1778	1790	1803	1815	1833	1940	1952	1965	1977	1995	2143	2155
	2166	2193	2320	2329	2338	2352	2366	2466	2475	2484	2498	2512	2612	2621	2630
	2644	2658	2758	2767	2776	2790	2804	2904	2913	2922	2936	2950	3082	3093	3104
	3115	3127	3138	3149	3160	3211	3223	3236	3248	3502	3512	3523	3623	3635	3648
	3660	3678	3898	3909	3928	3937	3946	3960	3974	4095	4106	4119	4257	4282	4289
	4324	4334	4368	4378	4404	4409	4414	4426	4430	4434	4440	4446	4451	4456	4462
	4467	4473	4478	4483	4488	4508	4518	4548	4581	4595	4606	4645	4659	4694	4715
	5674	5754	5757	5760	5765	5772	5820	5835	5856	5899	5916	5972	6004	6031	6054
	6060	6063	6076	6079	6139	6166	6168	6194	6217	6261	6287	6297	6306	6313	6349
	6363	6385	6431	6453	6558	6651	6661	6673	6710	6744	6770	6776	6780	6891	6896
	6946	6962	6965	6999	7016	7020	7024	7053	7080	7119	7131	7194	7211	7239	7245
	7252	7257	7282	7297	7325	7342	7356	7384	7462	7481	7486	7495	7592	7601	7610
	7615	7621	7627												
CLR	708	721	722	742	755	874	911	913	934	935	936	937	973	999	1055
	1130	1178	1246	1249	1315	1320	1385	1387	1398	1400	1439	1444	1543	1564	1579
	1636	1656	1697	1754	1773	1805	1854	1916	1935	1967	2016	2122	2157	2184	2214
	2253	2254	2292	2357	2399	2400	2438	2503	2545	2546	2584	2649	2691	2692	2730
	2795	2837	2838	2876	2941	2984	3029	3186	3187	3206	3238	3349	3377	3415	3460
	3552	3553	3598	3618	3650	3698	3799	3826	3862	3900	3965	4007	4052	4151	4152
	4205	4206	4287	4295	4315	4326	4336	4359	4391	4392	4498	4504	4505	4538	4570
	4599	4632	4643	4651	4683	4752	4770	4775	4776	4777	4795	4796	4797	4809	4865
	4883	4888	4889	4890	4908	4909	4910	4922	4978	4996	5001	5002	5003	5021	5022
	5023	5035	5091	5109	5114	5115	5116	5134	5135	5136	5148	5204	5222	5227	5228
	5229	5247	5248	5249	5261	5318	5330	5397	5479	5490	5493	5501	5523	5557	5579
	5599	5619	5639	5672	5689	5739	5740	5833	5891	5894	5997	6078	6093	6125	6126
	6337	6338	6437	6461	6511	6512	6596	6677	6756	6800	6809	6826	6828	6839	6858
	6899	6968	6977	7063	7100	7101	7105	7109	7117	7234	7276	7294	7318	7339	7355
	7408	7411	7445	7446	7459										
CLRB	1026	4657	4664	4665	4766	4784	4790	4879	4897	4903	4992	5010	5016	5105	5123
CMP	5129	5218	5236	5242	5920	6077	6222	6286	6312	6401	6402	6403	6887		
	709	732	752	901	1002	1009	1015	1021	1258	1330	1786	1799	1828	1847	1948
	1961	1990	2009	2104	2116	2139	2151	2179	2208	2286	2314	2325	2348	2432	2460
	2471	2494	2578	2606	2617	2640	2724	2752	2763	2786	2870	2898	2909	2932	3011
	3023	3045	3056	3067	3078	3089	3100	3111	3122	3134	3145	3156	3167	3219	3232
	3261	3331	3343	3355	3372	3383	3442	3454	3476	3487	3498	3508	3519	3530	3631
	3644	3673	3692	3781	3793	3821	3894	3922	3933	3956	4034	4046	4068	4079	4091

	4102	4115	4126	4329	4753	4830	4866	4943	4979	5056	5092	5169	5205	5282	5319
	5398	5491	5912	6061	6085	6109	6115	6135	6142	6154	6156	6186	6192	6195	6197
	6210	6515	6518	6529	6533	6539	6729	6810	6840	6966	7030	7107	7113	7115	7279
	7322	7352													
CMPB	899	1187	1470	1481	1567	1597	1640	1673	4373	4514	4553	4590	4611	4699	4747
	4860	4973	5086	5199	5325	5335	5404	5414	5484	5667	5751	5966	6067	6071	6117
	6149	6214	6220	6264	6279	6281	6269	6310	6314	6370	6462	6598	6605	6607	6616
	6621	6670	6674	6731	6749	6888	6913	7002	7004	7009	7056	7083	7235	7433	7472
COM	4803	4916	5029	5142	5255										
DEC	3576	3583	4175	4182	4229	4236	4268	4277	4398	4804	4810	4828	4917	4923	4941
	5030	5036	5054	5143	5149	5167	5256	5262	5280	5362	5450	5743	6005	6525	6658
	6715	6765	6768	6824	6843	6859	7420	7430	7443						
DECb	5841	5852	6293	6296	6645										
EMT	79														
HALT	61	4693	5975	6260	6430	6452	6771	7129	7256						
INC	751	893	974	1133	1181	1253	1324	1440	1445	1573	1648	3560	3568	4159	4167
	4213	4221	4355	5350	5438	5507	5524	5534	5564	5580	5600	5620	5640	5704	5741
	5847	5855	5898	5957	5984	6084	6164	6400	6438	6464	6538	6733	6762	6764	6872
	6890	6906	6960	6978	7011	7013	7061	7106	7118	7237	7243	7281	7324	7357	
INCB	1388	1402	1582	1602	1659	1679	4363	4502	4542	5951	6083	6316	6981	7032	7127
	7414														
IOT	80														
JMP	68	70	73	1752	1768	1914	1930	2086	2098	2108	2120	2131	2251	2265	2274
	2290	2301	2397	2411	2420	2436	2447	2543	2557	2566	2582	2593	2689	2703	2712
	2728	2739	2835	2849	2858	2874	2885	2982	2993	3002	3015	3027	3038	3049	3060
	3071	3184	3201	3314	3326	3335	3347	3359	3413	3424	3433	3446	3458	3469	3480
	3491	3596	3612	3763	3775	3785	3797	3808	3860	3873	3882	4005	4016	4025	4038
	4050	4061	4072	4083	4420	4749	4759	4862	4872	4975	4985	5088	5098	5201	5211
	5313	5392	5486	5680	5785	6549	6600	6737	6751	6783					
JSR	969	1433	1560	1575	1592	1633	1650	1668	1757	1772	1783	1785	1796	1798	1808
	1810	1820	1823	1827	1842	1845	1857	1859	1867	1919	1934	1945	1947	1958	1960
	1970	1972	1982	1985	1989	2004	2007	2019	2021	2029	2093	2103	2113	2115	2125
	2127	2136	2138	2149	2151	2160	2162	2172	2174	2176	2187	2189	2199	2201	2207
	2217	2219	2257	2270	2279	2281	2284	2286	2295	2297	2311	2313	2325	2334	2343
	2346	2348	2356	2360	2362	2371	2403	2416	2425	2427	2430	2432	2441	2443	2457
	2459	2471	2480	2489	2492	2494	2502	2506	2508	2517	2549	2562	2571	2573	2576
	2578	2587	2589	2603	2605	2617	2626	2635	2638	2640	2648	2652	2654	2663	2695
	2708	2717	2719	2722	2724	2733	2735	2749	2751	2763	2772	2781	2784	2786	2794
	2798	2800	2809	2841	2854	2863	2868	2870	2879	2881	2881	2895	2897	2909	2918
	2927	2930	2932	2940	2944	2946	2955	2998	3007	3009	3009	3020	3022	3032	3034
	3043	3045	3054	3056	3065	3067	3076	3078	3087	3087	3098	3100	3109	3111	3120
	3122	3132	3134	3143	3145	3154	3156	3165	3167	3190	3205	3216	3218	3229	3231
	3241	3243	3253	3256	3260	3271	3270	3274	3274	3285	3285	3285	3285	3285	3285
	3382	3418	3429	3438	3440	3451	3453	3463	3465	3474	3476	3485	3487	3496	3498
	3506	3508	3517	3519	3528	3530	3538	3560	3566	3568	3574	3576	3581	3583	3601
	3617	3628	3630	3641	3643	3653	3655	3665	3668	3672	3687	3690	3701	3703	3711
	3770	3780	3790	3792	3802	3804	3814	3816	3818	3829	3831	3865	3878	3886	3888
	3892	3894	3903	3905	3919	3921	3923	3942	3951	3954	3956	3965	3968	3970	3979
	4010	4021	4030	4032	4043	4045	4055	4057	4066	4068	4077	4079	4089	4091	4100
	4102	4113	4115	4124	4126	4157	4159	4165	4167	4173	4175	4180	4182	4211	4213
	4219	4221	4227	4229	4234	4236	4248	4254	4258	4267	4276	4318	4495	4499	4567
	4574	4600	4627	4636	4639	4653	4681	4687	4708	4821	4934	5047	5160	5273	5346
	5361	5434	5449	5648	5702	5712	5721	5780	5963	5969	6153	6269	6288	6295	6302
	6389	6481	6485	6740	6763	6798	6804	6842	6849	6969	6991	7034	7036	7038	7045

MOV	7072	7148	7152	7157	7161	7180	7183	7186	7200	7203	7217	7287	7295	7329	7340
	7377	7429	7436	7442	7475	7492	7527	7534							
	707	711	713	714	715	716	717	718	719	720	724	725	728	729	730
	731	736	738	739	740	745	748	749	750	757	759	766	767	768	771
	772	773	774	775	776	777	786	787	788	789	793	794	795	796	800
	801	802	803	807	808	809	810	814	815	816	817	821	822	823	824
	825	826	832	833	834	837	838	839	840	841	842	843	844	845	854
	855	856	857	858	859	860	861	862	863	864	877	879	881	883	885
	886	887	889	890	919	920	922	923	924	925	927	928	929	930	950
	951	967	972	1000	1007	1008	1014	1020	1062	1067	1072	1247	1316	1384	1395
	1396	1409	1431	1443	1472	1541	1544	1545	1546	1563	1626	1635	1779	1792	1820
	1823	1835	1845	1941	1954	1982	1985	1997	2007	2089	2110	2133	2145	2168	2169
	2177	2195	2196	2207	2281	2308	2313	2322	2343	2427	2454	2459	2468	2489	2573
	2600	2605	2614	2635	2719	2746	2751	2760	2781	2865	2892	2897	2906	2927	3004
	3017	3040	3084	3129	3212	3225	3253	3256	3317	3337	3354	3361	3362	3370	3382
	3435	3448	3471	3493	3514	3554	3624	3637	3665	3668	3680	3690	3766	3787	3810
	3811	3819	3889	3916	3921	3930	3951	4027	4040	4063	4086	4110	4153	4197	4207
	4256	4260	4266	4270	4272	4279	4280	4286	4308	4310	4313	4314	4316	4321	4322
	4327	4341	4354	4356	4365	4366	4371	4389	4390	4393	4400	4403	4407	4412	4422
	4425	4429	4433	4435	4439	4441	4445	4449	4454	4457	4461	4465	4468	4472	4474
	4477	4479	4482	4484	4487	4489	4496	4513	4519	4533	4535	4536	4545	4551	4569
	4578	4579	4587	4597	4603	4604	4609	4628	4629	4630	4631	4641	4642	4648	4655
	4656	4666	4679	4682	4698	4703	4704	4705	4711	4712	4719	4744	4751	4761	4762
	4764	4765	4768	4769	4771	4772	4773	4774	4779	4780	4782	4783	4786	4787	4788
	4791	4792	4793	4794	4799	4800	4801	4802	4807	4808	4813	4814	4815	4816	4817
	4818	4819	4825	4826	4827	4832	4833	4857	4864	4874	4875	4877	4878	4881	4882
	4884	4885	4886	4887	4892	4893	4895	4896	4899	4900	4901	4904	4905	4906	4907
	4912	4913	4914	4915	4920	4921	4926	4927	4928	4929	4930	4931	4932	4938	4939
	4940	4945	4946	4970	4977	4987	4988	4990	4991	4994	4995	4997	4998	4999	5000
	5005	5006	5008	5009	5012	5013	5014	5017	5018	5019	5020	5025	5026	5027	5028
	5033	5034	5039	5040	5041	5042	5043	5044	5045	5051	5052	5053	5058	5059	5083
	5090	5100	5101	5103	5104	5107	5108	5110	5111	5112	5113	5118	5119	5121	5122
	5125	5126	5127	5130	5131	5132	5133	5138	5139	5140	5141	5146	5147	5152	5153
	5154	5155	5156	5157	5158	5164	5165	5166	5171	5172	5196	5203	5213	5214	5216
	5217	5220	5221	5222	5224	5225	5226	5231	5232	5234	5235	5238	5239	5240	5243
	5244	5245	5246	5251	5252	5253	5254	5259	5260	5265	5266	5267	5268	5269	5270
	5271	5277	5278	5279	5284	5285	5314	5315	5317	5322	5323	5328	5329	5331	5332
	5333	5340	5341	5342	5343	5344	5345	5351	5352	5355	5356	5357	5358	5359	5360
	5393	5394	5396	5401	5402	5407	5408	5410	5411	5412	5419	5420	5421	5423	5424
	5424	5425	5426	5427	5428	5429	5430	5431	5432	5439	5440	5443	5444	5445	5446
	5447	5448	5483	5487	5488	5496	5497	5499	5500	5503	5504	5505	5506	5508	5517
	5518	5519	5520	5521	5522	5525	5526	5527	5528	5540	5541	5542	5543	5544	5545
	5546	5547	5548	5549	5555	5555	5556	5559	5560	5561	5562	5563	5573	5574	5575
	5576	5577	5578	5581	5582	5583	5584	5593	5594	5595	5596	5597	5598	5601	5602
	5603	5604	5613	5614	5615	5616	5617	5618	5621	5622	5623	5624	5625	5634	5635
	5636	5637	5638	5641	5642	5643	5644	5671	5678	5679	5682	5683	5684	5685	5686
	5687	5688	5690	5691	5692	5699	5700	5701	5705	5706	5707	5708	5709	5710	5711
	5714	5715	5716	5717	5718	5719	5720	5746	5768	5775	5777	5816	5824	5825	5826
	5832	5839	5857	5858	5859	5860	5861	5881	5882	5883	5884	5885	5886	5887	5892
	5895	5915	5921	5922	5923	5924	5925	5927	5928	5953	5958	5979	5982	5996	6001
	6010	6015	6020	6022	6026	6056	6057	6059	6062	6075	6087	6088	6091	6092	6095
	6096	6122	6146	6151	6180	6181	6208	6209	6224	6225	6226	6227	6262	6263	6268
	6276	6291	6330	6331	6332	6333	6334	6336	6351	6352	6353	6354	6355	6366	6367
	6374	6378	6383	6384	6386	6388	6398	6404	6405	6419	6420	6421	6422	6423	6424

	6425	6426	6427	6428	6429	6435	6436	6440	6441	6442	6443	6444	6445	6446	6447
	6448	6483	6503	6504	6505	6507	6509	6513	6514	6517	6520	6521	6527	6597	6603
	6611	6614	6615	6620	6623	6624	6625	6626	6627	6628	6630	6635	6638	6643	6646
	6647	6649	6654	6662	6664	6668	6680	6683	6686	6689	6694	6697	6700	6703	6704
	6706	6714	6717	6723	6728	6734	6748	6753	6754	6755	6758	6759	6760	6799	6805
	6807	6812	6813	6814	6815	6817	6818	6821	6829	6830	6831	6832	6833	6836	6845
	6846	6853	6854	6857	6864	6867	6868	6869	6870	6871	6874	6875	6876	6878	6879
	6881	6885	6895	6900	6903	6904	6918	6919	6921	6922	6923	6944	6959	6967	6976
	6980	6989	7029	7064	7097	7098	7099	7103	7110	7121	7122	7145	7146	7149	7150
	7153	7162	7177	7178	7184	7219	7233	7247	7272	7273	7275	7284	7286	7290	7291
	7293	7314	7315	7317	7327	7333	7334	7335	7338	7354	7406	7428	7432	7435	7440
	7441	7471	7479	7490	7526	7532	7552	7553	7557	7563	7564	7596	7598	7605	7607
	7618	7624													
MOVB	723	905	977	1001	1056	1077	1177	1248	1318	1386	1399	1468	1473	1551	1552
	1565	1572	1574	1590	1594	1595	1637	1647	1649	1666	1670	1671	4264	4294	4328
	4342	4372	4493	4512	4546	4552	4571	4588	4593	4610	4633	4684	4720	4767	4785
	4880	4898	4993	5011	5106	5124	5219	5237	5334	5413	5695	5696	5697	5698	5817
	5818	5821	5822	5823	5827	5830	5831	5850	5890	5893	5907	5910	5919	5960	5968
	6090	6094	6113	6130	6184	6190	6213	6218	6273	6301	6309	6339	6361	6362	6364
	6486	6604	6641	6644	6801	6886	6905	6908	6912	7028	7033	7035	7037	7042	7070
	7151	7154	7156	7158	7159	7160	7181	7182	7199	7201	7216	7218	7467	7530	7531
	7537	7555													
NEG	5828	5889													
NOP	918	1053	1054	5738	5781	5782	5783								
RESET	5779	7374													
ROL	5834	5836	5837	5838	5840	6342	6344	6346							
RTI	737	5862	5929	5986	6097	6152	6200	6228	6278	6356	6451	6865	6924	7421	7565
RTS	6024	6318	6406	6470	6487	6541	6850	7065	7085	7123	7163	7220	7248	7292	7336
	7360	7386	7413	7416	7447	7541	7558								
SUB	5489	5896	5959	6381	6531	6532	6735	6816							
TRAP	7567	7576	7577	7578	7579	7581	7583	7584	7585	7586					
TST	878	882	897	1547	1765	1774	1811	1860	1927	1936	1973	2022	2094	2127	2162
	2189	2219	2257	2270	2297	2334	2362	2403	2416	2443	2480	2508	2549	2562	2589
	2626	2654	2695	2708	2735	2772	2800	2841	2854	2881	2918	2946	2987	3198	3207
	3244	3322	3418	3609	3619	3656	3704	3771	3804	3831	3865	3878	3905	3942	3970
	4010	5673	5845	5901	5911	5973	5980	6028	6058	6082	6144	6159	6275	6283	6305
	6350	6376	6394	6396	6535	6547	6657	6819	6851	6855	7126	7277	7285	7319	7328
	7377	7404	7409	7554											
TSTB	1131	1139	1179	1251	1322	1558	4291	4319	4338	4361	4509	4540	4576	4601	4717
	5903	5917	6069	6111	6127	6182	6188	6258	6307	6368	6379	6392	6882	7058	7465
.ASCII	262	263	7767	7774	7827										
.ASCIZ	261	264	909	5756	5762	5767	5774	6032	6232	6233	6234	6236	6455	6653	6712
	6746	6778	6782	6964	7018	7022	7026	7254	7464	7488	7499	7594	7603	7612	7617
	7623	7630	7635	7641	7648	7653	7659	7666	7671	7676	7681	7686	7690	7695	7701
	7709	7713	7719	7721	7728	7735	7764	7786	7793	7797	7804	7808	7810	7812	7817
	7822	7835	7839	7844	7849	7859	7863	7867	7871	7873	7876	7879	7884	7888	7891
	7896	7899													
.ASECT	14														
.BLKB	6231														
.BLKW	487	5934	7915	8058	8078	8154	8220	8286	8352	8418	8484	8550	8616	8620	8621
	8622	8623	8624	8625											
.BYTE	231	232	237	238	246	247	255	256	257	258	280	281	291	292	299
	300	302	303	305	306	683	5787	5863	5864	5865	5866	5970	5971	6229	6230
	6407	6408	6409	6591	7967	7972	7976	7979	7984	7988	7991	7994	7999	8002	8008

8011	8016	8019	8028	8031	8036	8039	8044	8047	8052	8055	8100	8101	8103	8104
8106	8107	8109	8110	8112	8113	8115	8116	8118	8119	8121	8122	8124	8125	8128
8129	8131	8132	8135	8136	8141	8142	8166	8167	8169	8170	8172	8173	8175	8176
8178	8179	8181	8182	8184	8185	8187	8188	8190	8191	8194	8195	8197	8198	8201
8207	8207	8208	8232	8233	8235	8236	8238	8239	8241	8242	8244	8245	8247	8248
8250	8251	8253	8254	8256	8257	8260	8261	8263	8264	8267	8268	8273	8274	8278
8299	8301	8302	8304	8305	8307	8308	8310	8311	8313	8314	8316	8317	8319	8320
8322	8333	8326	8327	8329	8330	8333	8334	8339	8340	8364	8365	8367	8368	8370
8371	8373	8374	8376	8377	8379	8380	8382	8383	8385	8386	8388	8389	8392	8393
8395	8396	8399	8400	8405	8406	8430	8431	8433	8434	8436	8437	8439	8440	8442
8443	8445	8446	8448	8449	8451	8452	8454	8455	8458	8459	8461	8462	8465	8466
8471	8472	8496	8497	8499	8500	8502	8503	8505	8506	8508	8509	8511	8512	8514
8515	8517	8518	8520	8521	8524	8525	8527	8528	8531	8532	8537	8538	8562	8563
8565	8566	8568	8569	8571	8572	8574	8575	8577	8578	8580	8581	8583	8584	8586
8587	8590	8591	8593	8594	8597	8598	8603	8604						
.DSABL														
.ENABL														
.END														
.ENDC														
6169														
31	54	6102												
8640														
36	50	52	53	54	64	79	171	185	191	195	197	203	205	212
225	229	231	259	260	261	262	266	269	291	299	302	305	308	309
310	311	312	315	484	703	711	712	715	717	719	721	722	724	726
747	895	901	907	909	916	917	918	919	920	921	955	956	965	966
967	968	1138	1144	1186	1329	1393	1407	1414	1415	1429	1430	1431	1432	1461
1467	1480	1485	1496	1497	1539	1540	1541	1542	1699	1708	1709	1746	1747	1748
1779	1791	1804	1816	1834	1870	1871	1908	1909	1910	1941	1953	1966	1978	1996
2032	2033	2080	2081	2082	2144	2156	2167	2194	2221	2229	2230	2247	2248	2249
2306	2321	2330	2339	2353	2367	2375	2376	2393	2394	2395	2452	2467	2476	2485
2499	2513	2521	2522	2539	2540	2541	2598	2613	2622	2631	2645	2659	2667	2668
2685	2686	2687	2744	2759	2768	2777	2791	2805	2813	2814	2831	2832	2833	2890
2905	2914	2923	2937	2951	2960	2961	2977	2978	2979	3083	3094	3105	3116	3128
3139	3150	3161	3170	3176	3177	3178	3179	3212	3224	3237	3249	3263	3270	3271
3308	3309	3310	3385	3391	3392	3408	3409	3410	3503	3513	3524	3538	3539	3544
3545	3546	3549	3551	3588	3589	3590	3591	3624	3636	3649	3661	3679	3714	3715
3757	3758	3759	3833	3838	3839	3856	3857	3858	3899	3910	3914	3929	3938	3947
3961	3975	3983	3984	4000	4001	4002	4096	4107	4120	4128	4136	4137	4142	4143
4144	4148	4150	4189	4190	4195	4196	4197	4198	4202	4204	4241	4242	4245	4246
4247	4283	4290	4294	4299	4300	4306	4307	4308	4309	4325	4329	4335	4340	4343
4346	4347	4352	4353	4354	4367	4369	4373	4375	4379	4381	4382	4386	4387	4388
4509	4511	4519	4525	4526	4530	4531	4532	4547	4549	4553	4555	4559	4560	4564
4565	4566	4580	4582	4589	4596	4605	4607	4611	4613	4617	4618	4624	4625	4626
4643	4646	4657	4660	4672	4673	4677	4678	4679	4680	4692	4713	4716	4719	4721
4727	4728	4742	4743	4744	4745	4825	4830	4840	4841	4855	4856	4857	4858	4938
4943	4953	4954	4968	4969	4970	4971	5051	5056	5066	5067	5081	5082	5083	5084
5164	5169	5179	5180	5184	5195	5196	5197	5277	5282	5293	5294	5308	5309	5310
5349	5368	5369	5387	5388	5389	5437	5456	5457	5481	5482	5483	5484	5515	5571
5591	5611	5631	5653	5654	5664	5665	5666	5727	5728	5729	5730	5733	5734	5736
5739	5745	5748	5749	5756	5762	5767	5774	5777	5779	5785	5787	5788	5794	5871
5938	5941	5951	5958	5963	5964	5965	5973	5984	5987	5990	6005	6034	6037	6040
6045	6051	6053	6064	6067	6068	6069	6073	6080	6084	6084	6089	6091	6095	6098
6099	6102	6103	6105	6133	6169	6173	6201	6202	6209	6211	6214	6216	6232	6238
6244	6273	6323	6325	6358	6361	6362	6365	6392	6407	6418	6427	6428	6434	6440
6441	6451	6458	6653	6712	6746	6778	6782	6964	6994	7001	7018	7022	7026	7048
7055	7075	7082	7189	7196	7206	7213	7254	7448	7464	7488	7547	7553	7556	7575
7576	7577	7578	7579	7580	7581	7582	7583	7584	7585	7586	7587	7594	7603	7612

DRLPA.P11

CROSS REFERENCE TABLE

SEQ 0240

.EQUIV	7617 7679 163 269 6964	7623 80 164 909 7018 970	8628 88 165 5756 7022 1434	133 166 5762 7026	134 167 5767 7254	135 168 5774 7464	136 169 5788 7488	137 170 6033 7502	138 6410 7594	139 6457 7603	140 6653 7612	141 6712 7617	142 6746 7623	161 6778 7746	162 6782 7904
.GLOBL	14	970	1434												
.IDENT	7919														
.IFF	32 211 309 722 965 1466 1803 2082 2366 2597 2804 3082 3248 3529 3757 3982 4190 4306 4374 4552 4618 4718 4942 5178 5389 5666 5766 5964 6104 6322 6451 7074 7577 8528	50 224 310 724 967 1479 1815 2143 2374 2612 2812 3093 3262 3544 3759 3984 4195 4308 4374 4554 4624 4720 4952 5180 5426 5726 5773 5966 6105 6325 6455 7080 7578	51 228 311 742 968 1484 1833 2155 2376 2621 2814 3104 3269 3546 3832 4000 4197 4309 4380 4558 4626 4726 4954 5194 5455 5728 5777 5973 6067 6133 6337 6652 7188 7579	52 230 312 894 1137 1495 1869 2166 2393 2630 2831 3115 3271 3548 3837 4002 4198 4324 4382 4560 4642 4728 4970 5196 5481 5737 5787 5984 6070 6173 6360 6711 7194 7580	53 259 313 895 1143 1497 1871 2193 2395 2644 2833 3127 3308 3570 3839 4095 4201 4328 4386 4564 4645 4742 4970 5197 5483 5739 5785 5987 6070 6173 6365 6745 7205 7581	54 260 315 896 1185 1539 1908 2220 2451 2658 2839 3138 3310 3587 3856 4106 4203 4334 4388 4566 4656 4744 4971 5276 5483 5739 5787 5987 6071 6201 6365 6777 7211 7583	64 261 481 899 1328 1541 1910 2220 2466 2668 2839 3149 3384 3590 3859 4119 4240 4339 4508 4579 4655 4745 5055 5281 5484 5740 5788 5989 6080 6209 6392 6781 7253 7584	77 265 703 908 1392 1542 1940 2230 2475 2668 2833 3160 3390 3591 3858 4127 4242 4342 4510 4581 4673 4824 5055 5292 5484 5741 5789 6004 6082 6210 6407 6963 7424 7585	143 266 706 915 1406 1698 1952 2247 2484 2685 2839 3169 3392 3592 3862 4133 4247 4345 4511 4588 4673 4829 5065 5295 5487 5744 6020 6090 6214 6417 6993 7463 7586	171 268 711 917 1413 1707 1965 2249 2498 2687 2836 3175 3408 3593 3863 4137 4247 4347 4524 4595 4674 4839 5067 5298 5488 5745 6036 6092 6215 6427 6999 7487 7587	190 291 713 919 1415 1709 1977 2230 2498 2687 2836 3177 3410 3592 3864 4142 4282 4352 4526 4598 4674 4841 5081 5298 5490 5747 6039 6097 6231 6428 7017 7546 7593	193 299 715 920 1429 1746 1995 2220 2498 2687 2836 3179 3502 3650 3937 4144 4289 4354 4530 4606 4679 4855 5083 5298 5490 5747 6044 6098 6232 6433 7021 7552 7602	195 302 717 921 1431 1748 2031 2329 2522 2767 2961 3211 3512 3678 3946 4147 4293 4366 4532 4610 4680 4857 5084 5367 5498 5752 6054 6099 6238 6440 7025 7556 7611	202 305 719 954 1432 1778 2033 2338 2539 2776 2977 3223 3523 3671 3974 4149 4298 4368 4546 4612 4858 5163 5369 5498 5754 6051 6101 6243 6441 7047 7567 7616	204 308 721 956 1460 1790 2080 2352 2541 2790 2979 3236 3537 3715 3974 4188 4300 4372 4549 4616 4715 4937 5168 5387 5664 5963 6103 6264 6449 7053 7576 7622
.IFF	50 266 1144 1540 1910 2229 2467 2667 2905 3150 3310 3588	53 269 1186 1541 1941 2230 2476 2668 2914 3161 3385 3589	53 711 1329 1699 1953 2248 2485 2686 2923 3170 3391 3590	54 894 1393 1708 1966 2249 2499 2687 2937 3176 3392 3591	77 895 1407 1709 1978 2306 2513 2744 2951 3177 3409 3624	191 915 1414 1747 1996 2321 2521 2759 2960 3178 3410 3636	195 916 1415 1748 2032 2330 2522 2768 2961 3179 3503 3649	197 917 1430 1779 2033 2339 2540 2777 2978 3212 3513 3661	203 918 1431 1791 2081 2353 2541 2791 2979 3224 3524 3679	205 919 1461 1804 2082 2367 2549 2805 3083 3237 3538 3714	212 955 1467 1816 2144 2375 2613 2813 3094 3249 3539 3715	225 956 1480 1834 2156 2376 2622 2814 3105 3263 3545 3758	228 966 1485 1870 2167 2394 2631 2832 3116 3270 3546 3759	231 967 1496 1871 2194 2395 2645 2833 3128 3271 3549 3833	259 1138 1497 1909 2221 2452 2659 2890 3139 3309 3551 3838

	3839	3857	3858	3899	3910	3914	3929	3938	3947	3961	3975	3983	3984	4001	4002
	4096	4107	4120	4128	4136	4137	4143	4144	4148	4150	4189	4190	4196	4197	4198
	4202	4204	4241	4242	4246	4247	4283	4290	4294	4299	4300	4307	4308	4309	4325
	4328	4335	4340	4342	4346	4347	4353	4354	4367	4369	4372	4375	4379	4381	4382
	4387	4388	4509	4511	4519	4525	4526	4531	4532	4546	4549	4552	4555	4559	4560
	4565	4566	4580	4582	4588	4596	4605	4607	4610	4613	4617	4618	4625	4626	4643
	4646	4657	4660	4672	4673	4678	4679	4680	4692	4713	4716	4719	4720	4727	4728
	4743	4744	4745	4825	4830	4840	4841	4856	4857	4858	4938	4943	4953	4954	4969
	4970	4971	5051	5056	5066	5067	5082	5083	5084	5164	5169	5179	5180	5195	5196
	5197	5277	5282	5293	5294	5309	5310	5349	5368	5369	5388	5389	5437	5456	5457
	5482	5483	5484	5515	5570	5590	5610	5630	5653	5654	5665	5666	5727	5728	5729
	5730	5733	5735	5738	5745	5748	5787	5794	5871	5938	5940	5954	5984	5990	6005
	6034	6037	6064	6067	6068	6071	6098	6102	6105	6173	6175	6180	6201	6202	6211
	6215	6232	6244	6323	6361	6418	6434	6451	6993	6999	7047	7053	7074	7080	7188
	7194	7205	7211	7547	7553										
.IFT	909	5756	5762	5767	5774	5964	6079	6175	6180	6341	6357	6358	6653	6712	6746
.IFTF	6778	6782	6964	7018	7022	7026	7254	7464	7488	7594	7603	7612	7617	7623	7623
.IIF	909	5756	5762	5767	5774	5964	6079	6175	6180	6341	6357	6358	6653	6712	6746
	6746	6778	6782	6964	7018	7022	7026	7254	7464	7488	7594	7603	7612	7617	7623
	31	36	41	42	47	48	49	50	53	54	61	265	269	712	715
	721	722	724	725	895	5734	5739	5740	5769	5776	5787	5788	5941	5942	5943
	5944	5945	5950	5976	5984	5987	6002	6027	6040	6041	6042	6043	6044	6045	6049
	6078	6079	6095	6098	6099	6102	6123	6224	6232	6238	6320	6631	6655	6665	7480
.IRP	7575	7576	7577	7578	7579	7581	7583	7584	7585	7586	7619	7625			
	703	915	954	1413	1495	1707	1869	2031	2228	2374	2520	2666	2812	2959	3175
	3269	3390	3537	3587	3713	3837	3982	4135	4188	4240	4298	4345	4380	4524	4558
	4616	4671	4726	4839	4952	5065	5178	5292	5367	5455	5652	5726	5738	5881	5921
	5984	6050	6332	6353	6366	6367	6388	6404	6405	6421	6427	6440	6441		
.LIST	14	31	53	61	185	259	266	269	703	726	895	896	909	915	919
	954	967	1413	1431	1495	1541	1707	1748	1757	1772	1783	1785	1796	1798	1808
	1810	1820	1823	1827	1842	1845	1857	1859	1867	1869	1910	1919	1934	1945	1947
	1958	1960	1970	1972	1982	1985	1989	2004	2007	2019	2021	2029	2031	2082	2093
	2103	2113	2115	2125	2127	2136	2138	2149	2151	2160	2162	2172	2174	2176	2187
	2189	2199	2201	2207	2217	2219	2228	2249	2257	2270	2279	2281	2284	2286	2295
	2297	2311	2313	2325	2334	2343	2346	2348	2356	2360	2362	2371	2374	2395	2403
	2416	2425	2427	2430	2432	2441	2443	2457	2459	2471	2480	2489	2492	2494	2502
	2506	2508	2517	2520	2541	2549	2562	2571	2573	2576	2578	2587	2589	2603	2605
	2617	2626	2635	2638	2640	2648	2652	2654	2663	2666	2687	2695	2708	2717	2719
	2722	2724	2733	2735	2749	2751	2763	2772	2781	2784	2786	2794	2798	2800	2809
	2812	2833	2841	2854	2863	2865	2868	2870	2879	2881	2895	2897	2909	2918	2927
	2930	2932	2940	2944	2946	2955	2959	2979	2987	2998	3007	3009	3020	3022	3032
	3034	3043	3045	3054	3056	3065	3067	3076	3078	3087	3089	3098	3100	3109	3111
	3120	3122	3132	3134	3143	3145	3154	3156	3165	3167	3175	3179	3190	3205	3216
	3218	3229	3231	3241	3243	3253	3256	3260	3269	3310	3321	3330	3340	3342	3352
	3354	3365	3367	3369	3380	3382	3390	3410	3418	3429	3438	3440	3451	3453	3463
	3465	3474	3476	3485	3487	3496	3498	3506	3508	3517	3519	3528	3530	3537	3546
	3558	3560	3566	3568	3574	3576	3581	3583	3587	3591	3601	3617	3628	3630	3641
	3643	3653	3655	3665	3668	3672	3687	3690	3701	3703	3711	3713	3759	3770	3780
	3790	3792	3802	3804	3814	3816	3818	3829	3831	3837	3858	3865	3878	3886	3888
	3892	3894	3903	3905	3919	3921	3933	3942	3951	3954	3956	3965	3968	3970	3979
	3982	4002	4010	4021	4030	4032	4043	4045	4055	4057	4066	4068	4077	4079	4089
	4091	4100	4102	4113	4115	4124	4126	4135	4144	4157	4159	4165	4167	4173	4175
	4180	4182	4188	4197	4211	4213	4219	4221	4227	4229	4234	4236	4240	4247	4298
	4308	4345	4354	4380	4388	4524	4532	4558	4566	4616	4626	4671	4679	4726	4744

K03

.PAGE	222	315												
.PSECT	14													
.REM	1	14	31	499	535	566	585	980	1032	1080	1146	1194	1264	1336
.REPT	61	8102	8168	8234	8300	8366	8432	8498	8564		497	531	532	533
.SBTTL	43	55	75	188	200	222	266	315	495	496	497	661	705	891
	563	564	588	598	607	612	617	622	627	634	648	661	705	891
	915	954	968	979	1030	1079	1145	1192	1263	1335	1413	1494	1495	1703
	1869	2031	2228	2374	2520	2666	2812	2959	3175	3269	3390	3537	3587	3713
	3982	4135	4188	4240	4298	4345	4380	4524	4558	4616	4671	4726	4839	4952
	5178	5292	5367	5455	5652	5726	5730	5791	5868	5935	5987	6034	6099	6241
	6358	6415	6459	7544	7567	7629	7918							6320
.TITLE	31	7917												
.WORD	61	62	63	65	196	216	217	218	219	220	221	230	233	234
	236	239	240	241	242	243	244	245	248	249	250	271	272	273
	275	276	277	278	282	283	284	297	301	304	307	308	309	310
	312	464	467	469	471	473	475	477	479	481	482	484	486	491
	529	560	572	573	574	575	576	577	578	579	580	581	582	591
	595	597	600	601	602	603	604	608	609	610	613	614	615	618
	620	623	624	625	628	629	630	636	638	640	642	644	646	651
	653	654	655	656	657	658	663	664	665	666	667	668	669	670
	672	676	677	678	679	680	681	682	684	685	694	695	697	698
	700	701	971	1435	1700	1701	1757	1772	1783	1785	1796	1798	1808	1810
	1823	1827	1842	1845	1857	1859	1867	1919	1934	1945	1947	1958	1960	1970
	1982	1985	1989	2004	2007	2019	2021	2029	2093	2103	2113	2115	2125	2127
	2138	2149	2151	2160	2162	2172	2174	2176	2187	2189	2199	2201	2207	2217
	2257	2270	2279	2281	2284	2286	2295	2297	2311	2313	2325	2334	2343	2346
	2356	2360	2362	2371	2374	2416	2425	2427	2430	2432	2441	2443	2457	2459
	2480	2489	2492	2494	2502	2506	2508	2517	2549	2562	2571	2573	2576	2578
	2589	2603	2605	2617	2626	2635	2638	2640	2648	2652	2654	2663	2695	2708
	2719	2722	2724	2733	2735	2749	2751	2763	2772	2781	2784	2786	2794	2798
	2809	2841	2854	2863	2865	2868	2870	2879	2881	2895	2897	2909	2918	2927
	2932	2940	2944	2946	2955	2987	2998	3007	3009	3020	3022	3032	3034	3043
	3054	3056	3065	3067	3076	3078	3097	3089	3098	3100	3109	3111	3120	3122
	3134	3143	3145	3154	3156	3165	3167	3190	3205	3216	3218	3229	3231	3243
	3253	3256	3260	3321	3330	3340	3342	3352	3354	3365	3367	3369	3380	3382
	3429	3438	3440	3451	3453	3463	3465	3474	3476	3485	3487	3496	3498	3506
	3517	3519	3528	3530	3558	3560	3566	3568	3574	3576	3581	3583	3601	3617
	3630	3641	3643	3653	3655	3665	3668	3672	3687	3690	3701	3703	3711	3770
	3790	3792	3802	3804	3814	3816	3818	3829	3831	3865	3878	3886	3888	3892
	3903	3905	3919	3921	3933	3942	3951	3954	3956	3965	3968	3970	3979	4010
	4030	4032	4043	4045	4055	4057	4066	4077	4078	4079	4088	4091	4100	4102
	4115	4124	4126	4157	4159	4165	4167	4173	4175	4180	4182	4211	4213	4219
	4227	4229	4234	4236	5744	5747	5786	5867	6013	6018	6270	6317	6357	6390
	6471	6970	7067	7087	7222	7288	7289	7296	7331	7341	7362	7363	7377	7387
	7418	7422	7437	7476	7477	7482	7493	7494	7497	7498	7528	7529	7535	7536
	7574	7747	7749	7752	7753	7762	7909	7910	7911	7914	8059	8067	8068	8070
	8071	8072	8073	8074	8075	8079	8097	8098	8099	8102	8105	8108	8111	8114
	8120	8123	8127	8138	8139	8144	8145	8148	8149	8151	8163	8164	8165	8168
	8174	8177	8180	8183	8186	8189	8193	8204	8205	8210	8211	8214	8215	8217
	8230	8231	8234	8237	8240	8243	8246	8249	8252	8255	8259	8270	8271	8276
	8280	8281	8283	8295	8296	8297	8300	8303	8306	8309	8312	8315	8318	8321
	8336	8337	8342	8343	8346	8347	8349	8361	8362	8363	8366	8369	8372	8375
	8381	8384	8387	8391	8402	8403	8408	8409	8412	8413	8415	8427	8428	8429
	8435	8438	8441	8444	8447	8450	8453	8457	8468	8469	8474	8475	8478	8479

8493	8494	8495	8498	8501	8504	8507	8510	8513	8516	8519	8523	8534	8535	8540
8541	8544	8545	8547	8559	8560	8561	8564	8567	8570	8573	8576	8579	8582	8585
8589	8600	8601	8606	8607	8610	8611	8613							

000000

ERRORS DETECTED: 0

M03

DMDT MACY11 27(654) 13-DEC-77 12:58 PAGE 215
DRLPA.P11

SEQ 0245

*DRLPA,DRLPA/SOL/CRF=DRLPA.MAC,DRLPA
RUN-TIME: 40 34 5 SECONDS
CORE USED: 46K

DRLPXO (IMAGE) MICRO CODE
DRLPXO.P11

;DEFAULT TITLE MACY11 27(654) 13-DEC-77 13:01 PAGE 1

SEQ 0246

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

177777

000000'
000000

.....

.LIST MC,BIN,PEX,MEB
.NLIST MD,CND,ME

ADDRESS=-1
MACRO DEFFINITIONS FOR M800 AND M8204 MICR-PROCESSOR
INSTRUCTION SET.
TO BE USED WITH RSX MACRO-11 ASSEMBLER

26-MAY-1976
\$BEGIN
\$LOC 64000
.GLOBL MRCODE,UCODEE,IMAGE
.ENABL GBL

;*
;*MICRO CODE FOR KMC-11

;*THIS CODE WILL BE DOWN LOADED INTO BOTH
;*KMC-11'S. THE CODE RUNS ASYNCHRONOUS TO THE PDP-11 CODE
;*WE SYNC THROUGH COMMANDS PASSED VIA THE OUT*/IBUS* REGS.
;*

19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72

IMAGE: . JUMP TABLE USED FOR COMMANDS
BR STARTU ;GOTO START
.WORD .\$\$\$
BR CMNOP ;NOP=1
.WORD .\$\$\$
BR RDSILO ;=2 READ SILO PUT IN BSEL4
.WORD .\$\$\$
BR WRSILO ;=3 READ BSEL4 PUT IN SILO.
.WORD .\$\$\$
BR RDCMND ;=4 READ FAST PATH PUT IN BSEL4
.WORD .\$\$\$
BR WRCMND ;=5 READ BSEL4, PUT IN FAST PATH.
.WORD .\$\$\$
BR SAMP ;=6 TAKE AN A/D SAMPLE
.WORD .\$\$\$

; START OF U CODED

STARTU: MOVE # 0, BREG
.WORD .\$\$\$
MOVE BREG, OUT1 <0> ;CLEAR UNIBUS CSRS
.WORD .\$\$\$
MOVE BREG, OUT1 <2>
.WORD .\$\$\$
MOVE BREG, OUT1 <3>
.WORD .\$\$\$
MOVE BREG, OUT1 <4>
.WORD .\$\$\$
MOVE BREG, OUT1 <5>
.WORD .\$\$\$
MOVE BREG, OUT1 <6>
.WORD .\$\$\$
MOVE BREG, OUT1 <7>
.WORD .\$\$\$
MOVE BREG, SPAD <6>
.WORD .\$\$\$

CMNOP: MOVE INPD <12>, OUT1 <0> ;READ STATUS
.WORD .\$\$\$
MOVE # 377, BREG
.WORD .\$\$\$
MOVE BREG, OUT1 <2> ;INDICATE READY FOR COMMAND.
.WORD .\$\$\$

LOOP: MOVE INPD <12>, OUT1 <0> ;READ STATUS
.WORD .\$\$\$

064000
064000 100407
064002
064002 100420
064004
064004 100430
064006
064006 100432
064010
064010 100434
064012
064012 100436
064014
064014 100440

064016
064016 000400
064020
064020 061220
064022
064022 061222
064024
064024 061223
064026
064026 061224
064030
064030 061225
064032
064032 061226
064034
064034 061227
064036
064036 063226

064040
064040 021240
064042
064042 000777
064044
064044 061222

064046
064046 021240


```

73
74 064050          MOVE    INP1 <2>,SPAD <0> ;READ COMMAND REG.
75 064050 123040   .WORD   .SSS.
76 064052          BZ      LOOP          ;NO COMMAND THEN LOOP
77 064052 101423   .WORD   .SSS.
78
79 064054          MOVE    INP1 <2>,SPAD <0> ;RE-READ COMMAND.
80 064054 123040   .WORD   .SSS.
81
82 064056          BR      SPAD <0>          ;BR BASED ON CMND.
83 064056 160600   .WORD   .SSS.
84
85                ;NO-USER PROTECTION OFFERED.
86                ;IF YOU ENTER WRONG CODE -
87                ;YOU LOSE.
88
89                ;ROUTINE TO READ THE SILO, PUT IN
90                ;*BUS REG 4
91                ;CMD=2
92
93 RDSILO: MOVE     INP0 <10>,OUT1 <4> ;READ SILO.
94 064060 021204   .WORD   .SSS.
95                ;WRITE *BUS
96 064062          BR      CMNOP          ;RETURN.
97 064062 100420   .WORD   .SSS.
98
99                ;
100               ;ROUTINE TO WRITE SILO, READ DATA FROM
101               ;*BUS REG 4
102               ;CMD=3
103               ;
104
105 WRSILO: MOVE     INP1 <4>,OUT0 <10> ;READ DATA IN *BUS
106 064064 122110   .WORD   .SSS.
107                ;WRITE SILO.
108 064066          BR      CMNOP
109 064066 100420   .WORD   .SSS.
110
111               ;
112               ;ROUTINE TO READ FAST PATH (CMND) REG.
113               ;PUT IN *BUS REG 4
114               ;CMD=4
115               ;
116
117 RDCMND: MOVE     INP0 <11>,OUT1 <4> ;READ FAST PATH
118 064070 021224   .WORD   .SSS.
119                ;WRITE *BUS.
120 064072          BR      CMNOP          ;RETURN
121 064072 100420   .WORD   .SSS.
122
123               ;
124               ;ROUTINE TO WRITE FAST PATH (CMND) REG.
125               ;TAKE DATA FROM *BUS REG 4.
126               ;CMD=5

```

127
128
129 064074
130 064074 122111
131
132 064076
133 064076 100420
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149 064100
150 064100 020640
151 064102 103040
152 064104
153 064104 122151
154 064106
155 064106 020640
156 064110 103043
157 064112
158 064112 122111
159 064114
160 064114 020640
161 064116 103046
162 064120
163 064120 122131
164 064122
165 064122 020640
166 064124 103051
167 064126
168 064126 122171
169 064130
170 064130 020640
171 064132 103054
172 064134
173 064134 020640
174 064136 061620
175 064140 103056
176 064142
177 064142 020620
178 064144
179 064144 063220
180 064146

```

;
WRCMND: MOVE     INP1 <4>,OUTO <11> ;READ DATA IN *BUS
          .WORD   .$$$
          BR      CMNOP                ;WRITE INTO FAST PATH.
          .WORD   .$$$                ;RETURN.
    
```

```

; THIS ROUTINE TAKES AN A/D SAMPLE.
; CALL= CMND 6 IN BSEL2
; THESE REGS. MUST BE SET UP IN ADVANCE.
; BSEL 3 MUST CONTAIN READ CODE FOR A/D BUFFER.
; BSEL 4,5 MUST CONTAIN A/D CSR SETTING.
; BSEL 6 MUST CONTAIN WRITE CODE FOR A/D CSR
; BSEL 7 MUST CONTAIN READ CODE FOR A/D CSR
; BSEL 3,6,7 WILL REMAIN UNEFFECTED.
; BSEL 4,5 WILL CONTAIN A/D SAMPLE.
; BSEL2 WILL CONTAIN CODE 377 WHEN DONE.
    
```

```

WTMC     SAMP
          .WORD   .$$$
          .WORD   .$$$
          MOVE    INP1 <6>,OUTO <11>      ;SEND A/D WRITE CODE.
          .WORD   .$$$
          WTMC    SAMP1
          .WORD   .$$$
          .WORD   .$$$
          MOVE    INP1 <4>,OUTO <11>      ;SEND LOW BYTE CSR INFO.
          .WORD   .$$$
          WTMC    SAMP2
          .WORD   .$$$
          .WORD   .$$$
          MOVE    INP1 <5>,OUTO <11>      ;SEND HIGH BYTE CSR INFO.
          .WORD   .$$$
          WTMC    SLOOP
          .WORD   .$$$
          .WORD   .$$$
          MOVE    INP1 <7>,OUTO <11>      ;SEND READ CODE TO GET A/D CSR.
          .WORD   .$$$
          WTMC    SAMP3
          .WORD   .$$$
          .WORD   .$$$
          WTMM    SLOOP1
          .WORD   .$$$
          .WORD   .$$$
          .WORD   .$$$
          MOVE    INPO <11>,BREG
          .WORD   .$$$
          MOVE    BREG,SPAD <0>
          .WORD   .$$$
          WTMM    SLOOP2
    
```

181	064146	020640	.WORD	.SSS.	
182	064150	061620	.WORD	.SSS.	
183	064152	103063	.WORD	.SSS.	
184	064154		MOVE	INPD <11>,BREG	
185	064154	020620	.WORD	.SSS.	
186	064156		BB7	CMNOP	:ABORT IF A/D BIT 15=1
187	064156	103420	.WORD	.SSS.	
188	064160		MOVE	SPAD <0>,BREG	
189	064160	060600	.WORD	.SSS.	
190	064162		BB7	LOPE	
191	064162	103473	.WORD	.SSS.	
192	064164		BR	SLOOP	: IF A/D NOT DONE,EXIT.
193	064164	100451	.WORD	.SSS.	
194	064166		LOPE:	MOVE	INP1 <3>,OUT0 <11>
195	064166	122071	.WORD	.SSS.	:ISSUE READ A/B BUFFER.
196	064170		WTMM	SLOOP3	
197	064170	020640	.WORD	.SSS.	
198	064172	061620	.WORD	.SSS.	
199	064174	103074	.WORD	.SSS.	
200	064176		MOVE	INPD <11>,OUT1 <4>	
201	064176	021224	.WORD	.SSS.	
202	064200		WTMM	SLOOP4	
203	064200	020640	.WORD	.SSS.	
204	064202	061620	.WORD	.SSS.	
205	064204	103100	.WORD	.SSS.	
206	064206		MOVE	INPD <11>,OUT1 <5>	
207	064206	021225	.WORD	.SSS.	
208	064210		BR	CMNOP	
209	064210	100420	.WORD	.SSS.	
210	064212	177777	.WORD	-1	
211		000001	.END		

ADDRES= 177777	CLK = 000020	CMNOP 064040	IMAGE 064000 G
LOOP 064046	LOPE 064166	MARHLD= 000000	MARINC= 014000
MARLD = 010000	MARLDX= 004000	MRCODE= ***** G	PAGED = 000000
PAGE1 = 001000	PAGE2 = 002000	PAGE3 = 003000	PC = %000007
RDCMND 064070	RDSILO 064060	R0 = %000000	R1 = %000001
R2 = %000002	R3 = %000003	R4 = %000004	R5 = %000005
SAMP 064100	SAMP1 064106	SAMP2 064114	SAMP3 064130
SLOOP 064122	SLOOP1 064134	SLOOP2 064146	SLOOP3 064170
SLOOP4 064200	SP = %000006	STARTU 064016	UCODEE= ***** G
WRCMND 064074	WRSILO 064064	\$\$\$SER= 000001	.ADC = 000100
.ADD = 000000	.ADDWC= 000020	.AND = 000260	.BBD = 002000
.BB1 = 002400	.BB4 = 003000	.BB7 = 003400	.BC = 001000
.BR = 000400	.BSBRG= 160000	.BSIMM= 100000	.BSMEM= 140000
.BZ = 001400	.CO = 000400	.DBR = 000400	.DBRSH= 001400
.DEC = 000160	.DMEM = 002400	.DNOP = 000000	.DCUTO= 002000
.DOUT1= 001000	.DSPAD= 003000	.DSPBR= 003400	.DO = 000400
.FO = 000020	.INC = 000060	.LORN = 000240	.MINUS= 000260
.MO = 004000	.OR = 000300	.PLUS = 000000	.SBREG= 060000
.SELA = 000200	.SELB = 000220	.SIMM = 000000	.SINO = 020000
.SIN1 = 120000	.SMEM = 040000	.SUB = 000340	.SUBWC= 000040
.SUB2C= 000360	.SO = 020000	.XOR = 000320	.\$\$\$ = 100420
..LOC = 064040	.2A = 000120	.2AWC = 000140	. = 064214

ABCODE 000000 002
004000

ERRORS DETECTED: 0

GO4

DRLPX0 (IMAGE) MICRO CODE
DRLPX0.P11

;DEFAULT TITLE MACY11 27(654) 13-DEC-77 13:01 PAGE 7

SEQ 0252

*DRLPX0.DRLPX0/SOL=DRLPX0
RUN-TIME: 3 4 0 SECONDS
CORE USED: 6K

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

.LIST MC,BIN,BEX,MEB
.NLIST MD,CND,ME
.REM %

COPYRIGHT (C) 1975, 1976, 1977
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY
ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH
THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE
PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO
THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE
SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE
WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT
BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY
OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

%
.REM %

THIS CODE WAS DEVELOPED FOR USE WITH THE
LPA-11 DIAGNOSTIC, BY EDWARD C. BADGER.

%

177777

ADDRESS=-1
MACRO DEFFINITIONS FOR M800 AND M8204 MICR-PROCESSOR
INSTRUCTION SET.
TO BE USED WITH RSX MACRO-11 ASSEMBLER

26-MAY-1976

: THIS IS A MODIFIED VERSION OF THE MACROS WRITTEN BY W C BROWN
: MODIFIED FOR USE WITH MACY11 AND LNKX11.
: WARNING: "BAB" BITS TAKEN OUT OF BRANCH INSTRUCTIONS.

000000'
000000

\$BEGIN
\$LOC 65000
.GLOBL MRCODE,UCODEE,KMDIAG
.ENABL GBL

065000

:*
: *MICRO CODE FOR KMC-11
KMDIAG:

54
55
56
57
58
59
60
61
62
63
64 065000
65 065000 000777
66 065002
67 065002 061220
68 065004
69 065004 061222
70 065006
71 065006 061223
72 065010
73 065010 061224
74 065012
75 065012 061225
76 065014
77 065014 061226
78 065016
79 065016 061227
80
81 065020
82 065020 120400
83 065022
84 065022 101410
85
86
87
88
89
90
91
92 065024
93 065024 000400
94 065026
95 065026 061220
96 065030
97 065030 061222
98 065032
99 065032 061223
100 065034
101 065034 061224
102 065036
103 065036 061225
104 065040
105 065040 061226
106 065042
107 065042 061227

```

; THIS MICRO CODE WILL BE LOADED INTO THE
; KMC-11 VIA THE LPA-11 DIAG.
; ALL TESTS MUST BE EXECUTED IN ORDER BY THE
; PDP-11

; TEST #1
; CRAM ONES WRITE TEST
; ENTERED ON INIT.

TST1:  MOVE # 377, BREG ; GET ALL ONES
      .WORD .SSS.
      MOVE BREG, OUT1 <0> ; PUT ALL ONES INTO CRAM
      .WORD .SSS.
      MOVE BREG, OUT1 <2>
      .WORD .SSS.
      MOVE BREG, OUT1 <3>
      .WORD .SSS.
      MOVE BREG, OUT1 <4>
      .WORD .SSS.
      MOVE BREG, OUT1 <5>
      .WORD .SSS.
      MOVE BREG, OUT1 <6>
      .WORD .SSS.
      MOVE BREG, OUT1 <7>
      .WORD .SSS.

A1:    MOVE INP1 <0>, BREG ; WHEN PDP-11 DONE VERIFICATION
      .WORD .SSS.
      BZ A1 ; IT WILL CLEAR CARM ADDR. #0.
      .WORD .SSS.

; TEST #2
; CRAM ZEROS WRITE TEST
; ENTERED WHEN PDP-11 ZEROS CRAM ADDR. 0

TST2:  MOVE # 0, BREG ; GET ZERO
      .WORD .SSS.
      MOVE BREG, OUT1 <0> ; PUT INTO ALL CRAM
      .WORD .SSS.
      MOVE BREG, OUT1 <2>
      .WORD .SSS.
      MOVE BREG, OUT1 <3>
      .WORD .SSS.
      MOVE BREG, OUT1 <4>
      .WORD .SSS.
      MOVE BREG, OUT1 <5>
      .WORD .SSS.
      MOVE BREG, OUT1 <6>
      .WORD .SSS.
      MOVE BREG, OUT1 <7>
      .WORD .SSS.

```

```

108
109 065044          A2:  MOVE  INP1 <0>,BREG  ;GET ZERO
110 065044 120400   .WORD  .$$$
111 065046          BZ    TST3           ;WHEN =377, PDP-11 READY FOR NEXT TEST.
112 065046 101425   .WORD  .$$$
113 065050          BR    A2
114 065050 100422   .WORD  .$$$
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129 065052          TST3: MOVE  # 376,BREG      ;GET ZERO BIT 0
130 065052 000776   .WORD  .$$$
131 065054          BBO   T3E           ;ERROR IF BR.
132 065054 102062   .WORD  .$$$
133 065056          MOVE  # 375,BREG      ;BIT 1 = 0
134 065056 000775   .WORD  .$$$
135 065060          BBI   T3E           ;ERROR IF BR
136 065060 102462   .WORD  .$$$
137 065062          MOVE  # 357,BREG      ;BIT 4 = 0
138 065062 000757   .WORD  .$$$
139 065064          BBO   T3E           ;ERROR IF BR
140 065064 103062   .WORD  .$$$
141 065066          MOVE  # 177,BREG      ;BIT 7 = 0
142 065066 000577   .WORD  .$$$
143 065070          BBO   T3E           ;ERROR IF BR
144 065070 103462   .WORD  .$$$
145 065072          MOVE  # 1,BREG        ;POSITIVE BR TEST
146 065072 000401   .WORD  .$$$
147 065074          BBO   A3           ;SHOULD BRANCH
148 065074 102040   .WORD  .$$$
149 065076          BR    T3E           ;IF NOT, ERROR
150 065076 100462   .WORD  .$$$
151 065100          A3:  MOVE  # 2,BREG      ;BR ON BIT 1
152 065100 000402   .WORD  .$$$
153 065102          BBI   B3           ;SHOULD BR.
154 065102 102443   .WORD  .$$$
155 065104          BR    T3E           ;IF NOT, ERROR.
156 065104 100462   .WORD  .$$$
157 065106          B3:  MOVE  # 20,BREG       ;BR ON BIT 4
158 065106 000420   .WORD  .$$$
159 065110          BBO   C3           ;SHOULD BR.
160 065110 103046   .WORD  .$$$
161 065112          BR    T3E           ;IF NOT, ERROR.

```

```

: TEST #3
:   BRANCH TEST
:   ENTERED WHEN PDP-11 RETURNS 377 TO ADDR. 0
:
: NOTE WE HAVE TO ASSUME BR AND BZ WORK
: OR WE COULDN'T HAVE GOT THIS FAR.
: UPON SUCCESSFUL COMPLETION OF THIS TEST WE
: PUT CODE 377 INTO CRAM ADDR. 2
: AND 0 INTO CRAM ADDR. 0. ON FAILURE
: WE ONLY ZERO CRAM ADDR. 0.

```



```

162 065112 100462
163 065114
164 065114 000600
165 065116
166 065116 103451
167 065120
168 065120 100462
169 065122
170 065122 000400
171 065124
172 065124 101462
173 065126
174 065126 000777
175 065130
176 065130 101456
177 065132
178 065132 100462
179 065134
180 065134
181 065134 061222
182 065136
183 065136 100462
184 065140
185 065140 000400
186 065142
187 065142 061222
188 065144
189 065144
190 065144 000400
191 065146
192 065146 061220
193
194
195
196
197
198
199
200
201
202
203 065150
204 065150 120400
205 065152
206 065152 101467
207 065154
208 065154 100464
209
210 065156
211 065156 000400
212 065160
213 065160 061222
214 065162
215 065162 063224

C3: .WORD .$$$
      MOVE # 200,BREG ;BR IF BIT 7
      .WORD .$$$
      BB7 D3 ;SHOULD BR.
      .WORD .$$$
      BR T3E ;IF NOT, ERROR.
D3: .WORD .$$$
      MOVE # 0,BREG ;TEST BZ NEGATIVE
      .WORD .$$$
      BZ T3E ;ERROR IF BR.
      .WORD .$$$
      MOVE # 377,BREG ;POS BR
      .WORD .$$$
      BZ E3 ;SHOULD BR
      .WORD .$$$
      BR T3E ;ERROR IF NOT
      .WORD .$$$
E3: .WORD .$$$
      MOVE BREG,OUT1 <2> ;PUT 377 IN OUTPUT REG#2
      .WORD .$$$
      BR F3 ;IF BR INSTR FAILS TO WORK,
      .WORD .$$$
      MOVE # 0,BREG ;THEN A ZERO GETS PUT IN #2
      .WORD .$$$
      MOVE BREG,OUT1 <2> ;A SIGN OF AN ERROR.
      .WORD .$$$
F3: .WORD .$$$
      T3E: MOVE # 0,BREG ;SIGNAL PDP-11 THAT WE ARE
           .WORD .$$$
           MOVE BREG,OUT1 <0> ;THROUGH BR-TEST.
           .WORD .$$$

;TEST #4
;
;ALU TEST
;
;ENTERED WHEN PDP-11 PUTS A CODE 377 INTO
;CRAM ADDR 0
;ADDR 2 = 0 IF BAD
TST4: .WORD .$$$
      MOVE INP1 <0>,BREG ;GET CRAM ADDR 0
      .WORD .$$$
      BZ R1 ;WHEN = 377 DO TEST.
      .WORD .$$$
      BR TST4
      .WORD .$$$
A4: .WORD .$$$
      MOVE # 0,BREG ;GET = 0
      .WORD .$$$
      MOVE BREG,OUT1 <2>
      .WORD .$$$
      MOVE BREG,SPAD <4>
      .WORD .$$$

```

```

216 065164      DEC      SPAD <4>      ;MAKE = 377
217 065164 063164 .WORD    .$$$!.DSPAD
218 065166      BZ       B4           ;BR IF GOOD SEC.
219 065166 101475  .WORD    .$$$
220 065170      BR       T4E
221 065170 100503  .WORD    .$$$
222 065172      B4:     DEC      SPAD <4>      ;SEC = 376
223 065172 063164 .WORD    .$$$!.DSPAD
224 065174      INC      SPAD <4>      ;INC = 377
225 065174 063064 .WORD    .$$$!.DSPAD
226 065176      BZ       C4           ;BR IF = 377
227 065176 101501  .WORD    .$$$
228 065200      BR       T4E
229 065200 100503  .WORD    .$$$
230 065202      C4:     MOVE     # 377,BREG
231 065202 000777  .WORD    .$$$
232 065204      MOVE     BREG,OUT1 <2> ;RETURN GOOD CODE
233 065204 061222  .WORD    .$$$
234
235 065206      T4E:    MOVE     # 0,BREG      ;RETURN CODE 0
236 065206 000400  .WORD    .$$$
237 065210      MOVE     BREG,OUT1 <0>
238 065210 061220  .WORD    .$$$
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254 065212      TST5:   MOVE     INP1 <0>,BREG ;WAIT FOR PDP-11
255 065212 120400  .WORD    .$$$
256 065214      BZ       A5
257 065214 101510  .WORD    .$$$
258 065216      BR       TST5
259 065216 100505  .WORD    .$$$
260 065220      A5:     MOVE     # 2,BREG      ;SET TO DO NPR IN
261 065220 000402  .WORD    .$$$
262 065222      MOVE     BREG,OUT0 <5> ;SET HIGH ADDR.
263 065222 062225  .WORD    .$$$
264 065224      MOVE     # 124,BREG     ;GET LOW ADDR. OF ADDR. 1124
265 065224 000524  .WORD    .$$$
266 065226      MOVE     BREG,OUT0 <4> ;SET LOW ADDR.
267 065226 062224  .WORD    .$$$
268 065230      MOVE     # 1,BREG     ;SET TO nPR IN
269 065230 000401  .WORD    .$$$

```

```

;TEST 5
;
;NPR TEST
;
;ENTERED IN LINE WHEN PDP-11 PUTS CODE 377 INTO
;GRAM ADDR. 0.
;DOES AN INPUT DATA XFER FROM LOC 1124
;IN PDP-11 MEM TO LOC 1126
;DOES NO DATA CHECKING RETURNS
;CODE 0 TO GRAM ADDR. 0 WHEN DONE.
;

```

```

270 065232          MOVE      BREG,OUT1 <10> ;CAUSE NPR IN
271 065232 061230   .WORD      .$$$
272 065234          MOVE      INP1 <10>,BREG ;WAIT FOR NPR DONE.
273 065234 120600   .WORD      .$$$
274 065236          BBO      B5
275 065236 102116   .WORD      .$$$
276
277 065240          MOVE      INPO <0>,BREG ;GET LOW BYTE DATA.
278 065240 020400   .WORD      .$$$
279 065242          MOVE      BREG,OUTO <2> ;OUTPUT NPR DATA REG.
280 065242 062222   .WORD      .$$$
281 065244          MOVE      INPO <1>,BREG ;GET HIGH BYTE.
282 065244 020420   .WORD      .$$$
283 065246          MOVE      BREG,OUTO <3> ;SAVE FOR OUTPUT.
284 065246 062223   .WORD      .$$$
285 065250          MOVE      # 2,BREG ;GET OUTPUT ADDR. = 1126
286 065250 000402   .WORD      .$$$
287 065252          MOVE      BREG,OUTO <7>
288 065252 062227   .WORD      .$$$
289 065254          MOVE      # 126,BREG ;LOW PART
290 065254 000526   .WORD      .$$$
291 065256          MOVE      BREG,OUTO <6>
292 065256 062226   .WORD      .$$$
293 065260          MOVE      INP1 <11>,SPAD <4> ;/-STN-
294 065260 123224   .WORD      .$$$
295 065262          AND      SPAD <4>,BREG,SPAD
296 065262 063264   .WORD      .$$$
297 065264          MOVE      # 0,BREG
298 065264 000400   .WORD      .$$$
299 065266          OR      SPAD <4>,BREG,BREG
300 065266 060704   .WORD      .$$$
301 065270          MOVE      BREG,OUT1 <11>
302 065270 061231   .WORD      .$$$
303 065272          MOVE      # 21,BREG
304 065272 000421   .WORD      .$$$
305 065274          MOVE      BREG,OUT1 <10> ;SET NPR OUT.
306 065274 061230   .WORD      .$$$
307
308 065276          CS:      MOVE      INP1 <10>,BREG ;WAIT TILL DONE
309 065276 120600   .WORD      .$$$
310 065300          BBO      CS
311 065300 102137   .WORD      .$$$
312
313 065302          MOVE      # 0,BREG ;TELL PDP-11 WE'RE DONE.
314 065302 000400   .WORD      .$$$
315 065304          MOVE      BREG,OUT1 <0>
316 065304 061220   .WORD      .$$$
317
318
319
320
321
322
323
;
;TEST 6
;
;NPR TEST
;
;ENTERED IN LINE WHEN PDP 11 PUTS CODE 377 INTO

```

```

324      :      CRAM ADDR. 0.
325      :      DOES AN INPUT DATA XFER FROM LOC 1124
326      :      IN PDP-11 MEM TO LOC 1126
327      :      DOES NO DATA CHECKING RETURNS
328      :      CODE 0 TO CRAM ADDR. 0 WHEN DONE.
329      :
330 065306      TST6:  MOVE  INPI <0>,BREG  ;WAIT FOR PDP-11
331      .WORD  .SS$.
332 065310      BZ      A6
333 065310      .WORD  .SS$.
334 065312      BR      TST6
335 065312      .WORD  .SS$.
336
337 065314      A6:    MOVE  # 2,BREG      ;SET TO DO NPR IN
338      .WORD  .SS$.
339 065316      MOVE  BREG,OUTO <5> ;SET HIGH ADDR.
340      .WORD  .SS$.
341 065320      MOVE  # 124,BREG    ;GET LOW ADDR. OF ADDR. 1124
342      .WORD  .SS$.
343 065322      MOVE  BREG,OUTO <4> ;SET LOW ADDR.
344      .WORD  .SS$.
345 065324      MOVE  # 1,BREG      ;SET TO NPR IN
346      .WORD  .SS$.
347 065326      MOVE  BREG,OUT1 <10> ;CAUSE NPR IN
348      .WORD  .SS$.
349 065330      B6:    MOVE  INPI <10>,BREG ;WAIT FOR NPR DONE.
350      .WORD  .SS$.
351 065332      BBO
352 065332      .WORD  .SS$.
353
354 065334      MOVE  INPO <0>,BREG  ;GET LOW BYTE DATA.
355      .WORD  .SS$.
356 065336      MOVE  BREG,OUTO <2> ;OUTPUT NPR DATA REG.
357      .WORD  .SS$.
358 065340      MOVE  INPO <1>,BREG  ;GET HIGH BYTE.
359      .WORD  .SS$.
360 065342      MOVE  BREG,OUTO <3> ;SAVE FOR OUTPUT.
361      .WORD  .SS$.
362 065344      MOVE  # 2,BREG      ;GET OUTPUT ADDR. = 1126
363      .WORD  .SS$.
364 065346      MOVE  BREG,OUTO <7>
365      .WORD  .SS$.
366 065350      MOVE  # 126,BREG    ;LOW PART
367      .WORD  .SS$.
368 065352      MOVE  BREG,OUTO <6>
369      .WORD  .SS$.
370 065354      MOVE  INPI <11>,SPAD <4> ;/-STN-
371      .WORD  .SS$.
372 065356      AND   SPAD <4>,BREG,SPAD
373      .WORD  .SS$.
374 065360      MOVE  # 0,BREG
375      .WORD  .SS$.
376 065362      OR    SPAD <4>,BREG,BREG
377      .WORD  .SS$.

```

```

378 065364      MOVE      BREG,OUT1 <11> ;SET NPR OUT.
379 065364 061231 .WORD      .SS$.
380 065366      MOVE      # 21,BREG
381 065366 000421 .WORD      .SS$.
382 065370      MOVE      BREG,OUT1 <10>
383 065370 061230 .WORD      .SS$.
384
385 065372      C6:      MOVE      INP1 <10>,BREG ;WAIT TILL DONE
386 065372 120600 .WORD      .SS$.
387 065374      BBO      C6
388 065374 102175 .WORD      .SS$.
389
390 065376      MOVE      # 0,BREG ;TELL PDP-11 WE'RE DONE.
391 065376 000400 .WORD      .SS$.
392 065400      MOVE      BREG,OUT1 <0>
393 065400 061220 .WORD      .SS$.
394
395      ;TEST 7
396      :
397      :
398      :
399      :
400      :
401      :
402      :
403      :
404      :
405      :
406 065402      TST7:   MOVE      INP1 <0>,BREG ;WAIT TILL PDP-11 READY!
407 065402 120400 .WORD      .SS$.
408 065404      BZ      A7
409 065404 101604 .WORD      .SS$.
410 065406      BR      TST7
411 065406 100601 .WORD      .SS$.
412
413 065410      A7:      MOVE      INP1 <11>,SPAD <4>
414 065410 123224 .WORD      .SS$.
415 065412      AND      SPAD <4>,BREG,SPAD
416 065412 063264 .WORD      .SS$.
417 065414      MOVE      # 200,BREG
418 065414 000600 .WORD      .SS$.
419 065416      OR      SPAD <4>,BREG,BREG
420 065416 060704 .WORD      .SS$.
421 065420      MOVE      BREG,OUT1 <11> ;SET INTR TO ADDR. XX0
422 065420 061231 .WORD      .SS$.
423 065422      B7:      MOVE      INP1 <11>,BREG ;WAIT TILL DONE
424 065422 120620 .WORD      .SS$.
425 065424      BB7     B7
426 065424 103611 .WORD      .SS$.
427
428 065426      MOVE      # 0,BREG ;TELL PDP-11 WE INTERRUPTED
429 065426 000400 .WORD      .SS$.
430 065430      MOVE      BREG,OUT1 <0> ;IN CASE WE DIDN'T
431 065430 061220 .WORD      .SS$.

```

```

432
433 065432          C7:  MOVE  INP1 <0>,BREG  ;NOW WAIT FOR PDP-11 TO TELL
434 065432 120400   .WORD  .SSS.
435 065434          BZ     D7              ;US TO INTR. TO VECTOR XX4
436 065434 101620   .WORD  .SSS.
437 065436          BR     C7
438 065436 100615   .WORD  .SSS.
439
440 065440          D7:  MOVE  INP1 <11>,SPAD <4>
441 065440 123224   .WORD  .SSS.
442 065442          AND   SPAD <4>,BREG,SPAD
443 065442 063264   .WORD  .SSS.
444 065444          MOVE  # 300,BREG
445 065444 000700   .WORD  .SSS.
446 065446          OR    SPAD <4>,BREG,BREG
447 065446 060704   .WORD  .SSS.
448 065450          MOVE  BREG,OUT1 <11> ;SET INTR TO ADDR. XX4
449 065450 061231   .WORD  .SSS.
450
451 065452          E7:  MOVE  INP1 <11>,BREG  ;WAIT TILL DONE.
452 065452 120620   .WORD  .SSS.
453 065454          BB7   E7
454 065454 103625   .WORD  .SSS.
455
456 065456          MOVE  # 0,BREG          ;TELL PDP-11 WE THOUGHT
457 065456 000400   .WORD  .SSS.
458 065460          MOVE  BREG,OUT1 <0>    ;WE HAD INTERRUPTED!
459 065460 061220   .WORD  .SSS.
460
461 065462          BR
462 065462 100631   .WORD  .SSS.
463
464
465 065464 177777   .WORD  -1
466 000001 000001   .END

```

ADDRES= 177777	A1	065020	A2	065044	A3	065100	
A4	065156	A5	065220	A6	065314	A7	065410
B3	065106	B4	065172	B5	065234	B6	065330
B7	065422	CLK	= 000020	C3	065114	C4	065202
C5	065276	C6	065372	C7	065432	D3	065122
D7	065440	E3	065134	E7	065452	F3	065144
KMDIAG	065000	MARHLD=	000000	MARINC=	014000	MARLD	= 010000
MARLDX=	004000	MRCODE=	*****	PAGED	= 000000	PAGE1	= 001000
PAGE2	= 002000	PAGE3	= 003000	PC	=%000007	R0	=%000000
R1	=%000001	R2	=%000002	R3	=%000003	R4	=%000004
R5	=%000005	SP	=%000006	TST1	065000	TST2	065024
TST3	065052	TST4	065150	TST5	065212	TST6	065306
TST7	065402	T3E	065144	T4E	065206	UCODEE=	*****
\$\$\$SER=	000001	.ADC	= 000100	.ADD	= 000000	.ADDWC=	000020
.AND	= 000260	.B80	= 002000	.B81	= 002400	.B84	= 003000
.B87	= 003400	.BC	= 001000	.BR	= 000400	.BSBRG=	160000
.BSIMM=	100000	.BSMEM=	140000	.BZ	= 001400	.CO	= 000400
.DBR	= 000400	.DBRSH=	001400	.DEC	= 000160	.DMEM	= 002400
.DNOP	= 000000	.DOU10=	002000	.DOUT1=	001000	.DSPAD=	003000
.DSPBR=	003400	.DO	= 000400	.FO	= 000020	.INC	= 000060
.LORN	= 000240	.MINUS=	000360	.MO	= 004000	.OR	= 000300
.PLUS	= 000000	.SBREG=	060000	.SELA	= 000200	.SELB	= 000220
.SIMM	= 000000	.SINO	= 020000	.SIN1	= 120000	.SMEM	= 040000
.SUB	= 000340	.SUBWC=	000040	.SUB2C=	000360	.SO	= 020000
.XOR	= 000320	\$\$\$.	= 100631	..LOC	= 065462	.RA	= 000120
.ZAWC	= 000140	.	= 065466				

ABCODE 000000 002
004000

ERRORS DETECTED: 0

*DRLPX1,DRLPX1/SOL=DRLPX1
RUN-TIME: 8 8 0 SECONDS
CORE USED: 6K

W322 000000

LNKX11 V023 13-DEC-77 13:01

*DRLPA.LDA/B:0,DRLPA.MAP=DRLPA,DRLPX0,DRLPX1/E

LOAD MAP

IDENT: LPA.03

TRANSFER ADDRESS: 000001
LOW LIMIT: 000000
HIGH LIMIT: 004000

MODULE	DMDT	ADDRESS	SIZE
SECTION ENTRY			
(. ABS.)		000000	000000
D.OCQ		054247	
D.OCS		054224	
D.O&Q		054201	
D.OES		054156	
D.TCQ		054363	
D.TCQP		054477	
D.TCS		054340	
D.TCSP		054454	
D.TEQ		054315	
D.TEQP		054431	
D.TES		054272	
D.TESP		054406	
FRECOR		056246	
IMAGE		064000	
KMDIAG		065000	
()	000000	000000

MODULE	DRLPX0	ADDRESS	SIZE
SECTION ENTRY			
()	000000	000000
<ABCODE>		000000	004000

MODULE	DRLPX1	ADDRESS	SIZE
SECTION ENTRY			
()	004000	000000

UNDEFINED REFERENCES

- DRLPX0
- DRLPX1
- DRLPX2
- MRCODE
- UCODEE

0000000000 SECONDS

00010000

780223

F05
PDP10 411