

# DR11-C

DEVICE REGISTER TEST  
MD-11-DZDRC-G

EP-DZDRC-G-DL-A  
COPYRIGHT © 72-77  
FICHE 1 OF 1

AUG 1977  
**digital**  
MADE IN USA

This microfiche card contains a grid of frames. The frames are arranged in approximately 10 rows and 3 columns. Each frame contains a small table or list of data, which is too small to read clearly. The data appears to be organized in a structured format, possibly a register or test results table. The frames are separated by thin white lines, and the overall card has a dark, textured background.

B01

EJF1DCDCRBSEQ

00010000

770720

PDP10 411

: HDR1DZDRCGSEQ

00010000

770720

.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDRC-G-D  
PRODUCT NAME: DR11C DEVICE REGISTER TEST  
DATE RELEASED : APRIL, 1977  
MAINTAINER: DIAGNOSTIC GROUP  
AUTHOR: JOHN HITTELL

COPYRIGHT (C) DIGITAL EQUIPMENT CORPORATION  
1972, 1977

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

1. ABSTRACT

THIS IS A LOGIC TEST OF THE DR11C. FOR THIS TEST TO OPERATE  
A SPECIAL MAINTENANCE CABLE MUST BE CONNECTED (BCOBR).  
THIS TEST WILL CHECK UP TO 32 SEQUENTIAL DR11C'S.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 STANDARD COMPUTER

DR11C

BCOBR FOR EACH DR11C

2.2 STORAGE

2.2.1 PROGRAM STORAGE - THE ROUTINE USES MEMORY  
FROM 0000 TO 5200.

3. LOADING PROCEDURE

3.1 METHOD

ABSOLUTE LOADER

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTING

FOR SWITCHLESS PROCESSORS, THE PROGRAM WILL USE THE  
CONTENTS OF LOCATION 176 AS THE VALUE OF THE SWITCHES  
IF NO HARDWARE SWITCH REGISTER IS FOUND. THE PROGRAM  
ALLOWS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER  
THROUGH THE CONTROL G (IG) ROUTINE. LOCATION 174  
WILL BE USED AS THE SOFTWARE DISPLAY REGISTER.

STARTING AT SA 200 ALL SWITCHES SHOULD BE DOWN OR ZERO.  
(IF NOT ZERO, BIT 0 TO 8 WILL BE STARTING VECTOR.)

4.2 STARTING ADDRESS OR ADDRESSES

(A) 200 = START OF TEST--FOR NORMAL TESTING  
(B) 204 = SPECIAL ENTRANCE --FOR TESTING UNIQUE DR11C  
(C) 210 = RESTART--FOR STARTING AFTER SHUT DOWN

4.3 PROGRAM AND/OR OPERATOR ACTION

LOAD PROGRAM INTO MEMORY.

SET SWITCH REGISTER TO STARTING ADDRESS.  
LOAD ADDRESS.  
PRESS START.  
THE PROGRAM WILL STAY IN SECTION AND LOOP.

4.3.1 FOR SPECIAL ENTRANCE - SA204

1ST HALT SET SWITCH REGISTER EQUAL TO CSR ADDRESS OF DR11C  
PRESS CONTINUE  
2ND HALT SET SWITCH REGISTER EQUAL TO VECTOR ADDRESS OF DR11C  
PRESS CONTINUE  
RAISE SWITCH 10 TO "1" TO INHIBIT SEQUENCING TO NEXT DR11C

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

5.1.1 AT SA 200 .. THE INSTRUCTION AND LOGIC TEST.  
WITH ALL SWITCHES DOWN THE PROGRAM WILL PRINT  
OUT ON ERRORS AND CONTINUE IN TEST. (\* WILL  
BE PRINTED AT COMPLETION OF TESTING EACH DR11C)

5.1.2 SWITCH SETTINGS ARE

SW15 = 1 OR UP ... HALT ON ERROR  
SW14 = 1 OR UP ... SCOPE LOOP  
SW13 = 1 OR UP ... INHIBIT PRINTOUT  
SW12 = 1 OR UP ... NOT USED  
SW11 = 1 OR UP ... INHIBIT ITERATION LOOP  
SW10 = 1 OR UP ... DO NOT ADVANCE TO NEXT DR11C  
SW09 = 1 OR UP ... INHIBIT PRINTOUT OF DEVICE TESTED.

5.1.3

5.2. SUBROUTINE STRACTS

5.2.1 BEGIN SA 200

5.2.2 SCOPE

-----  
THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST  
IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING  
ADDRESS OF EACH SUB-TEST AS IT IS BEING ENTERED.  
IF A SCOPE LOOP IS REQUESTED, IT WILL JUMP TO THE  
START OF THE SUBTEST THAT THE SCOPE LOOP IS RE-  
QUESTED FOR.

5.2.3 HALT

-----  
IS A ROUTINE THAT PRINTS-OUT AN ADDRESS THAT TAGS  
THE FAILING SUBTEST, AND THE INCORRECT DATA AT  
THE TIME OF THE FAILURE. SEE 6.1

(5. OPERATING PROCEDURE CONT'D)

5.3 PROGRAM AND/OR OPERATOR ACTION

5.3.1 LOADING AND STARTING AT 200 WITH ALL SWITCHES DOWN IS THE INSTRUCTION AND LOGIC TEST. IF AN ERROR IS DETECTED HERE, THERE WILL BE A PRINTOUT. WHEN AN ERROR IS DETECTED AND IT IS NECESSARY TO SCOPE ON IT, PLACE SW15 L<sup>2</sup> TO HALT ON ERROR, THEN SW14 UP TO LOOP ON ERROR, THEN SW13 UP TO DELETE PRINTOUTS.

6. ERRORS

6.1 ERROR PRINTOUT

ARE IN A FOUR WORD FORMAT. THE 1ST IS THE PC+2 OF THE DETECTED ERROR. THE 2ND IS THE PROCESSOR STATUS REGISTER. THE 3RD IS DEVICE ADDRESS, THE 4TH IS VECTOR ADDRESS.

6.2 ERROR RECOVERY

DEPRESS CONTINUE TO RESTART SECTION

7. RESTRICTIONS

7.1 STARTING RESTRICTION

NONE

7.2 OPERATIONAL RESTRICTION

THE DR11C MUST HAVE THE BCOBR CABLE TO RUN THIS TEST.

NOTE THAT THE DR11C HAS FLOATING VECTORS:

THE BELOW IS THE ASSIGNMENT OF FLOATING VECTORS, THE ASSIGNED SEQUENCES ARE:

1. STARTING AT 300 AND WORKING UPWARD ALL DC11'S WILL BE ASSIGNED.
2. THEN ANY EXTRA KL11 CALLED FOR (VT05, VT06, LC11)
3. THEN ANY DP11 CALLED FOR.
4. THEN ANY DM11 CALLED FOR.
5. THEN ANY DN11 CALLED FOR.
6. THEN ANY DM11BB CALLED FOR.
7. THEN ANY DR11A CALLED FOR.
8. THEN ANY DR11C CALLED FOR.

THE DR11A AND DR11C DEVICE ADDRESSES WILL BE ASSIGNED IN THE USER AREA OF 767776 TO 764000. THE ASSIGNMENT OF ADDRESSES WILL START AT THE HIGH ADDRESS LIMIT AND PROCEED DOWNWARD. USERS AND SPECIAL SYSTEMS SHOULD START THEIR ASSIGNMENTS OF SPECIAL DEVICES AT THE LOW ADDRESS LIMIT AND WORK UP. AFTER ASSIGNING ALL DR11A'S, ASSIGN DR11C'S

767776 TO 767770	DR11C #0	;ASSUMING NO DR11A'S
767766 TO 767760	DR11C #1	
.		
767706 TO 767700	DR11C #7	
.		
767606 TO 767600	DR11C #15	

8. MISCELLANEOUS

WHERE THERE ARE MULTIPLE DR11C OR A SYSTEM AND IT IS DESIRED TO TEST ONLY ONE OF THEM. THIS MAY BE ACHIEVED BY USING THE SPECIAL STARTING ADDRESS AND PLACING SW10 ON A ONE (UP) TO INHIBIT SEQUENCING TO THE NEXT DR11C. SEE 4.3.1.

8.1 EXECUTION TIME

FOR EACH DR11C ABOUT 1 MINUTE

8.2 UNTESTED LOGIC

SIGNALS TO USER NOT TESTED:  
"NEW DATA READY"  
"DATA TRANSMITTED"  
"INIT" TO THE USER

9. PROGRAM DESCRIPTION

THIS PROGRAM WHEN STARTED AT 200 CHECKS THE STANDARD DR11-C'S

THE PROGRAM THEN PERFORMES AN INCREMENTAL LOGIC CHECK FOR THE SELECTED DR11C.

THE DATA REGISTER IS TESTED TO SEE IF "RESET" CLEARS IT, AND IF IT WILL HOLD ALL COMBINATIONS OF NUMBERS.

THE READ/WRITE BITS OF THE STATUS REGISTER ARE ALSO TESTED.

BOTH THE "A" AND "B" INTERRUPTS ARE TESTED TO SEE IF THEY INTERRUPT AT THE CORRECT BUS REQUEST LEVEL BR-5.

AT THE END OF THE TEST AN '\*' IS TYPED AND ALSO THE ADDRESSES OF THE DR11-C CONTROL STATUS REGISTER AND IT'S SIDE INTERRUPT VECTOR IS TYPED (IF SELECTED VIA SWITCH 9.). THE PROGRAM THEN RETESTS THE UNIT (IF SELECTED VIA SWITCH 10) OR SCANS TO THE NEXT DR11-C. IF ANOTHER DR11-C IS ON THE SYSTEM THEN THE PROGRAM RESTARTS TESTING

THE NEW DR11-C.

AFTER ALL DR11-C'S HAVE BEEN TESTED THE PROGRAM WILL  
TYPE 'END PASS' AND RESTART TESTING WITH THE INITIAL DR11-C.

IF NO ERRORS OCCUR AND THREE DR11-C'S ARE AVAILABLE  
AND SWITCH 9 IS DOWN THE PROGRAM WILL TYPE.

160000 770 \*  
157770 1000 \*  
157760 1010 \*

/

ETC.

IF SWITCH 9 IS UP THEN

\*  
\*  
\*

IF A POWER FAIL OCCURS THE PROGRAM WILL RESTART AT "START".



10. LISTING

%

```

296
297
298
299      177776
300      104000
301      167770
302
303      000000
304      000001
305      000002
306      000003
307      000004
308      000005
309      000006
310      000007
311
312
313
314      001000
315      002000
316      004000
317      020000
318      040000
319
320
321      .SBTTL  BASIC DEFINITIONS
322
323      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
324      001200      STACK= 1200
325      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
326      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
327
328      ;*MISCELLANEOUS DEFINITIONS
329      000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
330      000012      LF= 12      ;;CODE FOR LINE FEED
331      000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
332      000200      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED
333      177776      PS= 177776  ;;PROCESSOR STATUS WORD
334      .EQUIV  PS,PSW
335      177774      STKLMT= 177774 ;;STACK LIMIT REGISTER
336      177772      PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
337      177570      DSWR= 177570 ;;HARDWARE SWITCH REGISTER
338      177570      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
339
340      ;*GENERAL PURPOSE REGISTER DEFINITIONS
341      000000      R0= %0      ;;GENERAL REGISTER
342      000001      R1= %1      ;;GENERAL REGISTER
343      000002      R2= %2      ;;GENERAL REGISTER
344      000003      R3= %3      ;;GENERAL REGISTER

```

BASIC DEFINITIONS

345	000004	R4=	%4	:: GENERAL REGISTER
346	000005	R5=	%5	:: GENERAL REGISTER
347	000006	R6=	%6	:: GENERAL REGISTER
348	000007	R7=	%7	:: GENERAL REGISTER
349	000006	SP=	%6	:: STACK POINTER
350	000007	PC=	%7	:: PROGRAM COUNTER

.\*PRIORITY LEVEL DEFINITIONS

353	000000	PR0=	0	:: PRIORITY LEVEL 0
354	000040	PR1=	40	:: PRIORITY LEVEL 1
355	000100	PR2=	100	:: PRIORITY LEVEL 2
356	000140	PR3=	140	:: PRIORITY LEVEL 3
357	000200	PR4=	200	:: PRIORITY LEVEL 4
358	000240	PR5=	240	:: PRIORITY LEVEL 5
359	000300	PR6=	300	:: PRIORITY LEVEL 6
360	000340	PR7=	340	:: PRIORITY LEVEL 7

.\*"SWITCH REGISTER" SWITCH DEFINITIONS

363	100000	SW15=	100000	
364	040000	SW14=	40000	
365	020000	SW13=	20000	
366	010000	SW12=	10000	
367	004000	SW11=	4000	
368	002000	SW10=	2000	
369	001000	SW09=	1000	
370	000400	SW08=	400	
371	000200	SW07=	200	
372	000100	SW06=	100	
373	000040	SW05=	40	
374	000020	SW04=	20	
375	000010	SW03=	10	
376	000004	SW02=	4	
377	000002	SW01=	2	
378	000001	SW00=	1	
379		.EQUIV	SW09, SW9	
380		.EQUIV	SW08, SW8	
381		.EQUIV	SW07, SW7	
382		.EQUIV	SW06, SW6	
383		.EQUIV	SW05, SW5	
384		.EQUIV	SW04, SW4	
385		.EQUIV	SW03, SW3	
386		.EQUIV	SW02, SW2	
387		.EQUIV	SW01, SW1	
388		.EQUIV	SW00, SW0	

.\*DATA BIT DEFINITIONS (BIT00 TO BIT15)

391	100000	BIT15=	100000	
392	040000	BIT14=	40000	
393	020000	BIT13=	20000	
394	010000	BIT12=	10000	
395	004000	BIT11=	4000	
396	002000	BIT10=	2000	
397	001000	BIT09=	1000	
398	000400	BIT08=	400	
399	000200	BIT07=	200	
400	000100	BIT06=	100	

BASIC DEFINITIONS

401	000040	BIT05=	40
402	000020	BIT04=	20
403	000010	BIT03=	10
404	000004	BIT02=	4
405	000002	BIT01=	2
406	000001	BIT00=	1
407		.EQUIV	BIT09,BIT9
408		.EQUIV	BIT08,BIT8
409		.EQUIV	BIT07,BIT7
410		.EQUIV	BIT06,BIT6
411		.EQUIV	BIT05,BIT5
412		.EQUIV	BIT04,BIT4
413		.EQUIV	BIT03,BIT3
414		.EQUIV	BIT02,BIT2
415		.EQUIV	BIT01,BIT1
416		.EQUIV	BIT00,BIT0
417			
418		.*BASIC	"CPU" TRAP VECTOR ADDRESSES
419	000004	ERRVEC=	4
420	000010	RESVEC=	10
421	000014	TBITVEC=	14
422	000014	TRTVEC=	14
423	000014	BPTVEC=	14
424	000020	IOTVEC=	20
425	000024	PWRVEC=	24
426	000030	EMTVEC=	30
427	000034	TRAPVEC=	34
428	000060	TKVEC=	60
429	000064	TPVEC=	64
430	000240	PIRQVEC=	240
431		.ENABLE	ABS
432	000000	.=0	
433	000000	.+2	
434	000002	000000	HALT
435	000004	000006	.+2
436	000006	000000	HALT
437	000010	000012	.+2
438	000012	000000	HALT
439	000014	000016	.+2
440	000016	000000	HALT
441	000020	000022	.+2
442	000022	000000	HALT
443	000024	000026	.+2
444	000026	000000	HALT
445	000030	000032	.+2
446	000032	000000	HALT
447	000034	000036	.+2
448	000036	000000	HALT
449	000040	000042	.+2
450	000042	000000	HALT
451	000044	000046	.+2
452	000046	000000	HALT
453	000050	000052	.+2
454	000052	000000	HALT
455	000054	000056	.+2
456	000056	000000	HALT

```

: TIME OUT AND OTHER ERRORS
: RESERVED AND ILLEGAL INSTRUCTIONS
: "T" BIT
: TRACE TRAP
: BREAKPOINT TRAP (BPT)
: INPUT/OUTPUT TRAP (IOT) **SCOPE**
: POWER FAIL
: EMULATOR TRAP (EMT) **ERROR**
: "TRAP" TRAP
: TTY KEYBOARD VECTOR
: TTY PRINTER VECTOR
: PROGRAM INTERRUPT REQUEST VECTOR

```

L01

457	000060	000062	.+2
458	000062	000000	HALT
459	000064	000066	.+2
460	000066	000000	HALT
461	000070	000072	.+2
462	000072	000000	HALT
463	000074	000076	.+2
464	000076	000000	HALT
465	000100	000102	.+2
466	000102	000000	HALT
467	000104	000106	.+2
468	000106	000000	HALT
469	000110	000112	.+2
470	000112	000000	HALT
471	000114	000116	.+2
472	000116	000000	HALT
473	000120	000122	.+2
474	000122	000000	HALT
475	000124	000126	.+2
476	000126	000000	HALT
477	000130	000132	.+2
478	000132	000000	HALT
479	000134	000136	.+2
480	000136	000000	HALT
481	000140	000142	.+2
482	000142	000000	HALT
483	000144	000146	.+2
484	000146	000000	HALT
485	000150	000152	.+2
486	000152	000000	HALT
487	000154	000156	.+2
488	000156	000000	HALT
489	000160	000162	.+2
490	000162	000000	HALT
491	000164	000166	.+2
492	000166	000000	HALT
493	000170	000172	.+2
494	000172	000000	HALT
495	000174	000176	.+2
496	000176	000000	HALT
497	000200	000202	.+2
498	000202	000000	HALT
499	000204	000206	.+2
500	000206	000000	HALT
501	000210	000212	.+2
502	000212	000000	HALT
503	000214	000216	.+2
504	000216	000000	HALT
505	000220	000222	.+2
506	000222	000000	HALT
507	000224	000226	.+2
508	000226	000000	HALT
509	000230	000232	.+2
510	000232	000000	HALT
511	000234	000236	.+2
512	000236	000000	HALT

513	000240	000242	.+2
514	000242	000000	HALT
515	000244	000246	.+2
516	000246	000000	HALT
517	000250	000252	.+2
518	000252	000000	HALT
519	000254	000256	.+2
520	000256	000000	HALT
521	000260	000262	.+2
522	000262	000000	HALT
523	000264	000266	.+2
524	000266	000000	HALT
525	000270	000272	.+2
526	000272	000000	HALT
527	000274	000276	.+2
528	000276	000000	HALT
529	000300	000302	.+2
530	000302	000000	HALT
531	000304	000306	.+2
532	000306	000000	HALT
533	000310	000312	.+2
534	000312	000000	HALT
535	000314	000316	.+2
536	000316	000000	HALT
537	000320	000322	.+2
538	000322	000000	HALT
539	000324	000326	.+2
540	000326	000000	HALT
541	000330	000332	.+2
542	000332	000000	HALT
543	000334	000336	.+2
544	000336	000000	HALT
545	000340	000342	.+2
546	000342	000000	HALT
547	000344	000346	.+2
548	000346	000000	HALT
549	000350	000352	.+2
550	000352	000000	HALT
551	000354	000356	.+2
552	000356	000000	HALT
553	000360	000362	.+2
554	000362	000000	HALT
555	000364	000366	.+2
556	000366	000000	HALT
557	000370	000372	.+2
558	000372	000000	HALT
559	000374	000376	.+2
560	000376	000000	HALT
561	000400	000402	.+2
562	000402	000000	HALT
563	000404	000406	.+2
564	000406	000000	HALT
565	000410	000412	.+2
566	000412	000000	HALT
567	000414	000416	.+2
568	000416	000000	HALT

569	000420	000422	.+2
570	000422	000000	HALT
571	000424	000426	.+2
572	000426	000000	HALT
573	000430	000432	.+2
574	000432	000000	HALT
575	000434	000436	.+2
576	000436	000000	HALT
577	000440	000442	.+2
578	000442	000000	HALT
579	000444	000446	.+2
580	000446	000000	HALT
581	000450	000452	.+2
582	000452	000000	HALT
583	000454	000456	.+2
584	000456	000000	HALT
585	000460	000462	.+2
586	000462	000000	HALT
587	000464	000466	.+2
588	000466	000000	HALT
589	000470	000472	.+2
590	000472	000000	HALT
591	000474	000476	.+2
592	000476	000000	HALT
593	000500	000502	.+2
594	000502	000000	HALT
595	000504	000506	.+2
596	000506	000000	HALT
597	000510	000512	.+2
598	000512	000000	HALT
599	000514	000516	.+2
600	000516	000000	HALT
601	000520	000522	.+2
602	000522	000000	HALT
603	000524	000526	.+2
604	000526	000000	HALT
605	000530	000532	.+2
606	000532	000000	HALT
607	000534	000536	.+2
608	000536	000000	HALT
609	000540	000542	.+2
610	000542	000000	HALT
611	000544	000546	.+2
612	000546	000000	HALT
613	000550	000552	.+2
614	000552	000000	HALT
615	000554	000556	.+2
616	000556	000000	HALT
617	000560	000562	.+2
618	000562	000000	HALT
619	000564	000566	.+2
620	000566	000000	HALT
621	000570	000572	.+2
622	000572	000000	HALT
623	000574	000576	.+2
624	000576	000000	HALT

625	000600	000602	.+2
626	000602	000000	HALT
627	000604	000606	.+2
628	000606	000000	HALT
629	000610	000612	.+2
630	000612	000000	HALT
631	000614	000616	.+2
632	000616	000000	HALT
633	000620	000622	.+2
634	000622	000000	HALT
635	000624	000626	.+2
636	000626	000000	HALT
637	000630	000632	.+2
638	000632	000000	HALT
639	000634	000636	.+2
640	000636	000000	HALT
641	000640	000642	.+2
642	000642	000000	HALT
643	000644	000646	.+2
644	000646	000000	HALT
645	000650	000652	.+2
646	000652	000000	HALT
647	000654	000656	.+2
648	000656	000000	HALT
649	000660	000662	.+2
650	000662	000000	HALT
651	000664	000666	.+2
652	000666	000000	HALT
653	000670	000672	.+2
654	000672	000000	HALT
655	000674	000676	.+2
656	000676	000000	HALT
657	000700	000702	.+2
658	000702	000000	HALT
659	000704	000706	.+2
660	000706	000000	HALT
661	000710	000712	.+2
662	000712	000000	HALT
663	000714	000716	.+2
664	000716	000000	HALT
665	000720	000722	.+2
666	000722	000000	HALT
667	000724	000726	.+2
668	000726	000000	HALT
669	000730	000732	.+2
670	000732	000000	HALT
671	000734	000736	.+2
672	000736	000000	HALT
673	000740	000742	.+2
674	000742	000000	HALT
675	000744	000746	.+2
676	000746	000000	HALT
677	000750	000752	.+2
678	000752	000000	HALT
679	000754	000756	.+2
680	000756	000000	HALT

681	000760	000762	
682	000762	000000	
683	000764	000766	
684	000766	000000	
685	000770	000772	
686	000772	000000	
687	000774	000776	
688	000776	000000	
689		000020	
690	000020	004652	
691	000022	000340	
692	000024	005124	
693	000026	000340	
694	000030	004126	
695	000032	000340	
696	000034	005316	
697	000036	000340	
698		000046	
699	000046	004112	
700		000052	
701	000052	000000	
702		000174	
703	000174	000000	
704	000176	000000	
705		000200	
706	000200	000137	001250
707	000204	000137	001616
708	000210	000137	001726
709		001200	
710			
711			
712			
713	001200	167770	
714	001202	167772	
715	001204	167774	
716	001206	167773	
717	001210	000300	
718	001212	000302	
719	001214	000304	
720			
721			
722			
723	001216	177570	
724	001220	177570	
725	001222	167770	
726	001224	167772	
727	001226	167774	
728	001230	167773	
729			
730	001232	000300	
731	001234	000302	
732	001236	000304	
733	001240	000000	
734			
735	001242	000000	
736	001244	000240	

```

.+2
HALT
.+2
HALT
.+2
HALT
.+2
HALT
.+2
HALT
.=20
.SCOPE
340
PFAIL
340
.HLT
340
TYP
340
.=46
$ENDAD
.=52
0
.=174
DISPRE: 0
SWREG: 0
.=200
JMP @#START1 ;INITIAL START
JMP @#SPEC ;TO SELECT UNIQUE ADDRESS AND VECTOR
JMP @#START ;RESTART
.=1200

```

;THIS TABLE CONTAINS INITIAL REGISTER AND VECTOR ADDRESSES

```

RCSR: CSR
      CSR+2
      CSR+4
      CSR+3
RCSR1: 300
        302
        304

```

;THIS TABLE CONTAINS REGISTER AND VECTOR ADDRESSES OF THE DR11-C UNDER TEST

```

SWR: 177570
DISPLA: 177570
DRCSR: 167770 ;ADDRESS OF DR11-C STATUS REGISTER
DROBUF: 167772 ;ADDRESS OF DR OUTPUT BUFFER REG.
DRIBUF: 167774 ;ADDRESS OF DR INPUT BUFFER REG.
DRBHIO: 167773 ;HIGH BYTE OF OUTPUT BUFFER REG.

DRVECA: 300 ;INTERRUPT VECTOR OF UNIT UNDER TEST
DRLVL: 302
DRVECB: 304 ;INTERRUPT VECTOR
XORFLG: 0

COUNT: 0 ;COUNT LOCATION
PL: 240 ;PRICRITY LEVEL

```



```

737 001246 000000 PASCNT: 0 ;NUMBER OF PASSES COMPLETED
738
739 001250 START1:
740 .SBTTL INITIALIZE THE COMMON TAGS
741 001250 012706 001200 MOV #STACK, SP ;SETUP THE STACK POINTER
742 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
743 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
744 001254 013746 000004 MOV @ERRVEC, -(SP) ;SAVE ERROR VECTOR
745 001260 012737 001314 000004 MOV #64$, @ERRVEC ;SET UP ERROR VECTOR
746 001266 012767 177570 177722 MOV #OSWR, SWR ;SETUP FOR A HARDWARE SWICH REGISTER
747 001274 012767 177570 177716 MOV #DISP, DISPLAY ;AND A HARDWARE DISPLAY REGISTER
748 001302 022777 177777 177706 CMP #-1, @SWR ;TRY TO REFERENCE HARDWARE SWR
749 001310 001012 BNE 66$ ;BRANCH IF NO TIMEOUT TRAP OCCURRED
750 ;AND THE HARDWARE SWR IS NOT = -1
751 001312 000403 BR 65$ ;BRANCH IF NO TIMEOUT
752 001314 012716 001322 64$: MOV #65$, (SP) ;SET UP FOR TRAP RETURN
753 001320 000002 RTI
754 001322 012767 000176 177666 65$: MOV #SWREG, SWR ;POINT TO SOFTWARE SWR
755 001330 012767 000174 177662 MOV #DISPREG, DISPLAY
756 001336 012637 000004 66$: MOV (SP)+, @ERRVEC ;RESTORE ERROR VECTOR
757
758 001342 023737 000042 000046 CMP @#42, @#46
759 001350 001433 BEQ 3$ ;YES, SKIP TITLE
760 .SBTTL TYPE PROGRAM NAME
761 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
762 001352 005227 177777 INC #-1 ;FIRST TIME?
763 001356 001027 BNE 67$ ;BRANCH IF NO
764 001360 104401 001416 TYPE 68$ ;TYPE ASCIZ STRING
765 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
766 001364 005737 000042 TST @#42 ;ARE WE RUNNING UNDER XXDP/ACT?
767 001370 001006 BNE 69$ ;BRANCH IF YES
768 001372 026727 177620 000176 CMP SWR, #SWREG ;SOFTWARE SWITCH REG SELECTED?
769 001400 001005 BNE 70$ ;BRANCH IF NO
770 001402 104405 GTSWR ;GET SOFT-SWR SETTINGS
771 001404 000403 BR 70$
772 001406 112767 000001 005144 69$: MOV #1, SAUTOB ;SET AUTO-MODE INDICATOR
773 001414 70$:
774 001414 000410 BR 67$ ;GET OVER THE ASCIZ
775 ;;68$: .ASCIZ <CRLF>#MD-11-DZDRC-G<CRLF>
776 001436 67$:
777 001436 000463 BR 8$ ;SKIP INTERRUPT VECTOR SIZER IF
778 ;IN AUTOMATIC MODE
779 ;SIZE FOR INTERRUPT VECTOR IF IN AUTOMATIC MODE.
780
781 001440 012700 000302 3$: MOV #302, R0 ;SET UP FLOATING VECTOR AREA
782 001444 010060 177776 4$: MOV R0, -2(R0)
783 001450 012720 000003 MOV #3, (R0)+
784 001454 005720 TST (R0)+
785 001456 022700 000776 CMP #776, R0
786 001462 100370 BPL 4$
787 001464 012737 001516 000014 MOV #55, @#14 ;SET UP BPT
788 001472 012737 000340 000016 MOV #340, @#16 ;SET UP BPT PSW
789 001500 012777 000101 177472 MOV #101, @RCR ;FORCE AN INTERRUPT
790 001506 000240 NOP
791 001510 000240 NOP
792 001512 000000 HALT ;NO INTERRUPT OCCURRED

```

```

793 001514 000423          BR      6$
794 001516 162716 000004    5$: SUB    #4 (SP)          ;FIGURE OUT INTERRUPT VECTOR
795 001522 011637 001210    MOV    (SP), 2#RCSR1    ;PUT IT IN TABLE
796 001526 013737 001210 001212    MOV    2#RCSR1, 2#RCSR1+2
797 001534 062737 000002 001212    ADD    #2, 2#RCSR1+2    ;FILL IN NEXT TABLE ENTRY
798 001542 013737 001210 001214    MOV    2#RCSR1, 2#RCSR1+4
799 001550 062737 000004 001214    ADD    #4, 2#RCSR1+4    ;FILL IN LAST TABLE ENTRY
800 001556 012737 000016 000014    MOV    #16, 2#14        ;RESTORE BPT
801 001564 012700 000302    6$: MOV    #302, R0      ;SET UP TRAP CATCHER
802 001570 010060 177776    7$: MOV    R0, -2(R0)
803 001574 005020          CLR    (R0)+
804 001576 005720          TST   (R0)+
805 001600 022700 000776    CMP    #776, R0
806 001604 100371          BPL   7$
807 001606 004767 000020    8$: JSR   PC, FIRST
808 001612 000137 001726    JMP   2#START
809 001616 012706 001200    SPEC: MOV   #STACK, %6
810 001622 004767 000004    JSR   PC, FIRST
811 001626 000167 003360    JMP   SPEC0
812 001632 013746 000004    FIRST: MOV  2#4, -(%6)
813 001636 012737 001712 000004    MOV   #XORA, 2#4
814 001644 012737 000031 177060    MOV   #31, 2#177060
815 001652 012637 000004    MOV   (%6)+, 2#4
816 001656 012737 177777 001240    MOV   #-1, 2#XORFLG
817 001664 012701 160000    MOV   #160000, R1
818 001670 004737 005246    JSR   PC, 2#SPEC1
819 001674 012701 000770    MOV   #770, R1
820 001700 004737 005276    JSR   PC, 2#SPEC2
821 001704 104401          TYPE
822 001706 006640          MESS1
823 001710 000207          RTS   PC
824 001712 022626    XORA: CMP   (%6)+, (%6)+
825 001714 012637 000004    MOV   (%6)+, 2#4
826 001720 005037 001240    CLR   2#XORFLG
827 001724 000207          RTS   PC
828
829 001726 012700 001200    ;INITIALIZE ADDRESS AND VECTORS
830 001732 012701 001222    START: MOV  #RCSR, R0    ;GET ADDRESS OF FIRST POSSIBLE DR11-C'S
831 001736 012021          MOV   #DRCSR, R1
832 001740 012021          MOV   (R0)+, (R1)+    ;LOAD INITIAL TEST ADDRESSES
833 001742 012021          MOV   (R0)+, (R1)+
834 001744 012021          MOV   (R0)+, (R1)+
835 001746 012021          MOV   (R0)+, (R1)+
836 001750 012021          MOV   (R0)+, (R1)+
837 001752 012021          MOV   (R0)+, (R1)+
838 001754 012706 001200    RSTART: MOV  #STACK, %6    ;SET UP STACK
839 001760 012767 002010 003004    MOV   #BEGIN, RETURN    ;SET SCOPE RETURN
840 001766 005037 004770    CLR   2#SCOPEF
841
842          ;DOES RESET CLEAR REGISTER?
843 001772 104406          CKSWR
844 001774 032777 001000 177214    BIT   #SW9, 2#SWR
845 002002 001002          BNE   BEGIN
846 002004 004737 005010    JSR   PC, 2#MOREID
847 002010 016705 177206    BEGIN: MOV  #DRCSR, R5    ;GET ADDRESS OF STATUS REGISTER
848 002014 012777 000240 175754    MOV   #240, 2#PSW      ;SET PRIORITY LEVEL 6

```

849	002022	012737	002060	000004		MOV	#15, R4	;SET TIME OUT TRAP VECTOR
850	002030	012767	000010	002730		MOV	#10, ICOUNT	
851	002036	012777	177777	177160		MOV	#-1, DROBUF	;PRESET OUTPUT BUFFER
852	002044	000005				RESET		;CLEAR DATA REGISTER
853	002046	017700	177152			MOV	DROBUF, %0	;GET RESULT OF RESET
854	002052	001403				BEQ	25	
855	002054	104000				HLT		;DATA REGISTER NOT CLEAR
856	002056	000401				BR	25	
857	002060	104000			15:	HLT		;ERROR! TIMED OUT WHEN REFERENCING DROBUF.
858	002062	012706	001200		25:	MOV	#STACK, SP	;RESET STACK POINTER
859	002066	012737	000006	000004		MOV	#6, R4	;RESTORE TIME OUT TRAP
860								
861	002074	000004				SCOPE		
862	002076	012767	004000	002662		MOV	#4000, ICOUNT	
863	002104	012777	177777	177112		MOV	#-1, DROBUF	;ALL ONES TO REGISTER
864	002112	017700	177106			MOV	DROBUF, %0	
865	002116	022700	177777			CMP	#-1, %0	
866	002122	001401				BEQ	+.4	
867	002124	104000				HLT		;REG WILL NOT HOLD ONES
868								
869	002126	000004				SCOPE		
870	002130	012767	000010	002630		MOV	#10, ICOUNT	
871	002136	012777	177777	177060		MOV	#-1, DROBUF	
872	002144	000005				RESET		;SET DATA TO ALL ONES
873	002146	005777	177054			TST	DROBUF	;SHOULD CLEAR REGISTER
874	002152	001401				BEQ	+.4	
875	002154	104000				HLT		;REG FAILED TO CLEAR
876								
877	002156	000004				SCOPE		
878	002160	012767	004000	002600		MOV	#4000, ICOUNT	
879	002166	012777	052525	177030		MOV	#52525, DROBUF	
880	002174	017700	177024			MOV	DROBUF, %0	
881	002200	022700	052525			CMP	#52525, %0	
882	002204	001401				BEQ	+.4	
883	002206	104000				HLT		;DATA NOT=52525
884								
885	002210	000004				SCOPE		
886	002212	012777	125252	177004		MOV	#125252, DROBUF	
887	002220	017700	177000			MOV	DROBUF, %0	
888	002224	022700	125252			CMP	#125252, %0	
889	002230	001401				BEQ	+.4	
890	002232	104000				HLT		;DATA NOT=125252
891								
892								
893	002234	000004				SCOPE		
894	002236	012737	000040	004766		MOV	#40, ICOUNT	
895	002244	010502			BUFTST:	MOV	R5, R2	;GET ADDRESS OF DRCSR
896	002246	005722				TST	(R2)+	;R2=ADDRESS OF OUTPUT BUFFER REG.
897	002250	012703	000401			MOV	#401, R3	;LOAD CONSTANT
898	002254	012704	000400		15:	MOV	#256, R4	;SET COUNT
899	002260	005000				CLR	R0	;PRESET EXPECTED RESULT
900	002262	005012				CLR	(R2)	;CLEAR REGISTER
901	002264	060300			25:	ADD	R3, R0	
902	002266	060312				ADD	R3, (R2)	
903	002270	021200				CMP	(R2), R0	
904	002272	001401				BEQ	+.4	

905	002274	104000				HLT		
906	002276	005304				DEC	R4	
907	002300	001371				BNE	2S	
908	002302	006303				ASL	R3	
909	002304	001363				BNE	1S	
910								
911								;TEST THAT BYTE REFERENCE TO DROBUF AFFECT PROPER BYTE ONLY
912								
913	002306	000004				SCOPE		
914	002310	012777	177777	176706	TAG:	MOV	#-1, @DROBUF	
915	002316	105077	176702			CLRB	@DROBUF	;CLEAR LOW BYTE
916	002322	017700	176676			MOV	@DROBUF,%0	
917	002326	022700	177400			CMP	#177400,%0	
918	002332	001401				BEQ	.+4	
919	002334	104000				HLT		;BYTE LOW FAILED TO CLEAR
920								
921	002336	000004				SCOPE		
922	002340	012777	177777	176656		MOV	#-1, @DROBUF	
923	002346	105077	176656			CLRB	@DRBHIO	;CLEAR HIGH BYTE
924	002352	017700	176646			MOV	@DROBUF,%0	
925	002356	022700	000377			CMP	#377,%0	
926	002362	001401				BEQ	.+4	
927	002364	104000				HLT		;HIGH BYTE CLEAR FAILED
928								
929	002366	000004				SCOPE		
930	002370	005037	002440			CLR	@2S	
931	002374	012704	002440			MOV	#2S,R4	
932	002400	005077	176620			CLR	@DROBUF	
933	002404	105077	176620			CLRB	@DRBHIO	
934	002410	105277	176614		1S:	INCB	@DRBHIO	;INCREMENT HIGH BYTE
935	002414	105264	000001			INCB	1(R4)	
936	002420	027714	176600			CMP	@DROBUF,(F)	
937	002424	001401				BEQ	.+4	
938	002426	104000				HLT		;HIGH BYTE HAS BAD DATA
939	002430	105764	000001			TSTB	1(R4)	
940	002434	001402				BEQ	3S	
941	002436	000764				BR	1S	
942	002440	000000					0	
943	002442	000004						
944								
945	002444	005015				CLR	(R5)	
946	002446	011500				MOV	(R5),R0	
947	002450	001401				BEQ	.+4	
948	002452	104000				HLT		
949	002454	012715	000140			MOV	#140,@R5	;INTERRUPT ENABLE FOR A+B
950	002460	011500				MOV	@R5,%0	
951	002462	022700	000140			CMP	#140,%0	;ENABLE BITS
952	002466	001401				BEQ	.+4	
953	002470	104000				HLT		
954								
955	002472	000004				SCOPE		
956	002474	012767	000010	002264		MOV	#10,ICOUNT	
957	002502	012715	000140			MOV	#140,@R5	;SET INTERRUPT ENABLE FLOPS
958	002506	000005				RESET		;CLEAR THOSE FLOPS
959	002510	011500				MOV	@R5,%0	
960	002512	001401				BEQ	.+4	

;CONTROL STATUS REGISTER (DRCSR) TESTS.



1017	002720	000004			SCOPE	
1018	002722	012777	177777	176274	MOV	#-1, @DROBUF
1019	002730	017777	176272	176266	MOV	@DRIBUF, @DROBUF ; MOV ALL ONES
1020	002736	022777	177777	176260	CMP	#-1, @DROBUF
1021	002744	001401			BEQ	.+4
1022	002746	104000			HLT	; NOT ALL ONES
1023						
1024	002750	000004			SCOPE	
1025	002752	005067	002010		CLR	ICOUNT
1026	002756	005000			CLR	%0
1027	002760	010077	176240		MOV	%0, @DROBUF ; TEST ALL NUMBERS
1028	002764	017777	176236	176232	MOV	@DRIBUF, @DROBUF
1029	002772	020077	176226		CMP	%0, @DROBUF
1030	002776	001401			BEQ	.+4
1031	003000	104000			HLT	; ERROR - CHECK %0 FOR GOOD
1032	003002	005200			INC	%0 ; DROBUF FOR BAD
1033	003004	001403			BEQ	TST9
1034	003006	005077	176212		CLR	@DROBUF
1035	003012	000762			BR	TST6
1036	003014	000004			TST9:	SCOPE
1037	003016	012737	000005	004766	MOV	#5, @ICOUNT
1038					; TEST DATA FROM BLACK BOX (NOT CONNECTED)	
1039	003024	012777	177777	176172	MOV	#-1, @DROBUF
1040	003032	017777	176166	176166	MOV	@DROBUF, @DRIBUF ; STATIC LINES EQUAL ONES
1041	003040	017700	176162		MOV	@DRIBUF, %0 ; DATA REGISTER TO %0
1042	003044	022700	177777		CMP	#-1, %0
1043	003050	001401			BEQ	.+4
1044	003052	104000			HLT	; REG 0 SHOULD = ALL ONES
1045						
1046					; READY BIT IS IN A ONE STATE	
1047	003054	000004			SCOPE	
1048	003056	012715	000003		MOV	#3, @RS ; CS0 AND CS1
1049	003062	011500			MOV	(RS), R0
1050	003064	022700	100203		CMP	#100203, R0
1051	003070	001401			BEQ	.+4
1052	003072	104000			HLT	
1053						
1054					; CAN WE RAISE INTERRUPT "A"	
1055	003074	000004			SCOPE	
1056	003076	052737	000340	177776	BIS	#340, @PSW ; LOCK OUT INTERRUPTS
1057	003104	012706	001200		MOV	#STACK, %6
1058	003110	012777	003132	176114	MOV	#TST4, @DRVECA ; INTERRUPT RETURN POINTER
1059	003116	012715	000101		MOV	#101, @RS ; INTERRUPT ENABLE AND CS0
1060	003122	005037	177776		CLR	@PSW
1061	003126	000240			NOP	
1062	003130	104000			HLT	; NO "A" INTERRUPT
1063	003132	005015			TST4:	@RS
1064	003134	016777	176074	176070	MOV	@RVL, @DRVECA ; MOVE .+2 TO "A" INTERRUPT VECTOR
1065						
1066					; RAISE INTERRUPT "B"	
1067	003142	000004			SCOPE	
1068	003144	012706	001200		MOV	#STACK, %6
1069	003150	052737	000340	177776	BIS	#340, @PSW
1070	003156	012777	003202	176052	MOV	#TST5, @DRVECB
1071	003164	012715	000042		MOV	#42, @RS ; IE AND CS1
1072	003170	042737	000377	177776	BIC	#377, @PSW

```

1073 003176 000240      NOP
1074 003200 104000      HLT
1075 003202 005015      TST5: CLR      JRS      ;NO B INTERRUPT
1076
1077      ;TEST FOR INTERRUPT FROM DEVICE
1078 003204 016777 176034 176022      MOV      PL, JORLVI
1079 003212 042737 000340 177776      BIC      #340, J#PSW      ;PROCESSOR LEVEL ZERO
1080 003220 012777 003252 176004      MOV      #TINT1, JORVECA
1081 003226 012706 001200      MOV      #STACK, %6      ;STACK POINTER
1082 003232 042777 000100 175762      BIC      #100, JORCSR      ;CLEAR INTERRUPT ENABLE
1083 003240 052777 000101 175754      BIS      #101, JORCSR      ;SET INTERRUPT ENABLE-AND CSRO
1084 003246 000240      NOP
1085 003250 104000      HLT
1086 003252 000004      TINT1: SCOPE      ;NO DEVICE INTERRUPT OCCURED
1087
1088      ;TEST FOR INTERRUPT FROM THE DEVICE
1089 003254 042737 000340 177776      BIC      #340, J#PSW
1090 003262 052737 000040 177776      BIS      #040, J#PSW      ;SET TO PRIORITY LEVEL 1
1091 003270 012777 003322 175734      MOV      #TINT2, JORVECA      ;INTERRUPT VECTOR ADDRESS
1092 003276 012706 001200      MOV      #STACK, %6      ;SET UP STACK POINTER
1093 003302 042777 000100 175712      BIC      #100, JORCSR      ;CLEAR INTERRUPT ENABLE
1094 003310 052777 000101 175704      BIS      #101, JORCSR      ;SET INTERRUPT ENABLE-AND CSRO
1095 003316 000240      NOP
1096 003320 104000      HLT
1097      ;NO DEVICE INTERRUPT OCCURED
1098 003322 000004      TINT2: SCOPE
1099 003324 042737 000340 177776      BIC      #340, J#PSW
1100 003332 052737 000100 177776      BIS      #100, J#PSW      ;SET TO PRIORITY LEVEL 2
1101 003340 012777 003372 175664      MOV      #TINT3, JORVECA      ;INTERRUPT VECTOR ADDRESS
1102 003346 012706 001200      MOV      #STACK, %6      ;SET UP STACK POINTER
1103 003352 042777 000100 175642      BIC      #100, JORCSR      ;CLEAR INTERRUPT ENABLE
1104 003360 052777 000101 175634      BIS      #101, JORCSR      ;SET INTERRUPT ENABLE-AND CSRO
1105 003366 000240      NOP
1106 003370 104000      HLT
1107      ;NO DEVICE INTERRUPT OCCURED
1108 003372 000004      TINT3: SCOPE
1109      ;TEST FOR INTERRUPT FROM THE DEVICE
1110 003374 042737 000340 177776      BIC      #340, J#PSW
1111 003402 052737 000140 177776      BIS      #140, J#PSW      ;SET TO PRIORITY LEVEL 3
1112 003410 012777 003442 175614      MOV      #TINT4, JORVECA      ;INTERRUPT VECTOR ADDRESS
1113 003416 012706 001200      MOV      #STACK, %6      ;SET UP STACK POINTER
1114 003422 042777 000100 175572      BIC      #100, JORCSR      ;CLEAR INTERRUPT ENABLE
1115 003430 052777 000101 175564      BIS      #101, JORCSR      ;SET INTERRUPT ENABLE-AND CSRO
1116 003436 000240      NOP
1117 003440 104000      HLT
1118 003442 000004      TINT4: SCOPE      ;NO DEVICE INTERRUPT OCCURED
1119
1120      ;TEST FOR INTERRUPT FROM DEVICE
1121 003444 042737 000340 177776      BIC      #340, J#PSW
1122 003452 052737 000200 177776      BIS      #200, J#PSW      ;RAISE PROCESSOR PRIORITY TO LEVEL 4
1123 003460 012777 003522 175544      MOV      #TINT5, JORVECA      ;IN CASE OF INTERRUPT
1124 003466 012706 001200      MOV      #STACK, %6      ;SET STACK POINTER
1125 003472 042777 000100 175522      BIC      #100, JORCSR      ;CLEAR INTERRUPT ENABLE
1126 003500 052777 000101 175514      BIS      #101, JORCSR      ;SET INTERRUPT ENABLE AND CSRO
1127 003506 000240      NOP
1128 003510 042777 000100 175504      BIC      #100, JORCSR      ;LET INTERRUPT OCCUR

```

```

1129 003516 000240      NOP
1130 003520 104000      HLT          ;NO DEVICE INTERRUPT OCCURED
1131 003522 000004      TINT5: SCOPE
1132
1133      ;TEST FOR NO INTERRUPT FROM DEVICE (HIGHEST PROCESSOR PRIORITY)
1134 003524 052737 000340 177776      BIS      #340, @PSW      ;RAISE PROCESSOR PRIORITY TO HIGHEST LEVEL
1135 003532 012777 003572 175472      MOV      @TINT6, @DRVECA ;IN CASE OF INTERRUPT
1136 003540 012706 001200      MOV      @STACK, %5     ;SET STACK POINTER
1137 003544 042777 000100 175450      BIC      #100, @DRCSR   ;CLEAR INTERRUPT ENABLE
1138 003552 052777 000101 175442      BIS      #101, @DRCSR
1139 003560 000240      NOP
1140 003562 042777 000100 175432      BIC      #100, @DRCSR
1141 003570 000401      BR      .+4             ;WITH NO INTERRUPT, BRANCH OVER HALT
1142 003572 104000      TINT6: HLT             ;INTERRUPT OCCURED
1143 003574 000004      SCOPE
1144
1145      ;TEST FOR NO INTERRUPT FROM DEVICE
1146 003576 042737 000340 177776      BIC      #340, @PSW
1147 003604 052737 000240 177776      BIS      #240, @PSW      ;RAISE PROCESSOR PRIORITY TO LEVEL 5
1148 003612 012777 003652 175412      MOV      @TINT7, @DRVECA ;IN CASE OF INTERRUPT
1149 003620 012706 001200      MOV      @STACK, %5     ;SET STACK POINTER
1150 003624 042777 000100 175370      BIC      #100, @DRCSR   ;CLEAR INTERRUPT ENABLE
1151 003632 052777 000101 175362      BIS      #101, @DRCSR   ;SET INTERRUPT ENABLE AND CSRO
1152 003640 000240      NOP
1153 003642 042777 000100 175352      BIC      #100, @DRCSR   ;DON'T LEAVE IT SET
1154 003650 000401      BR      .+4             ;WITH NO INTERRUPT, BRANCH OVER HALT
1155 003652 104000      TINT7: HLT             ;INTERRUPT OCCURED
1156 003654 000004      SCOPE
1157
1158      ;TEST FOR NO INTERRUPT FROM DEVICE
1159 003656 042737 000340 177776      BIC      #340, @PSW
1160 003664 052737 000300 177776      BIS      #300, @PSW      ;RAISE PROCESSOR PRIORITY TO LEVEL 6
1161 003672 012777 003732 175332      MOV      @TINT8, @DRVECA ;IN CASE OF INTERRUPT
1162 003700 012706 001200      MOV      @STACK, %5     ;SET STACK POINTER
1163 003704 042777 000100 175310      BIC      #100, @DRCSR   ;CLEAR INTERRUPT ENABLE
1164 003712 052777 000101 175302      BIS      #101, @DRCSR   ;SET INTERRUPT ENABLE-AND CSRO
1165 003720 042777 000100 175274      BIC      #100, @DRCSR   ;DON'T LEAVE IT SET
1166 003726 000240      NOP
1167 003730 000401      BR      .+4             ;WITH NO INTERRUPT, BRANCH OVER HALT
1168 003732 104000      TINT8: HLT             ;INTERRUPT OCCURED
1169 003734 000004      SCOPE
1170
1171 003736 016777 175272 175266      MOV      DR1VL, @DRVECA ;FOR FALSE INTERRUPT
1172 003744 005077 175262      CLR      @DRVECA
1173
1174      ;END OF TEST ROUTINE
1175 003750 012777 000052 000356      END: MOV      #'*, @TOBR ;TYPE '*'
1176 003756 105777 000354      1$: TSTB     @TCSR
1177 003762 100375      BPL     1$
1178 003764 005077 000344      CLR     @TOBR
1179 003770 105777 000342      2$: TSTB     @TCSR
1180 003774 100375      BPL     2$
1181 003776 104406      CKSWR
1182 004000 032777 002000 175210      BIT     #SW10, @SWR     ;LOOP ON SELECTED DR?
1183 004006 001402      BEQ     4$
1184 004010 000137 001754      JMP     @RSTART        ;REPEAT TEST ON DR11C SELECTED

```



```

1185 ;STEP TO NEXT DR11-C
1186 004014 012700 000010 4S: MOV #10,R0 ;STEPPING CONSTANT
1187 004020 012737 004072 000004 MOV #55,@#4 ;SET TIME OUT TRAP
1188 004026 160005 SUB RO,R5 ;STEP TO NEXT DR11-C ADDRESS
1189 004030 005715 TST (R5) ;WILL TIME OUT IF NOT AVAILABLE
1190 004032 012705 001222 MOV #DRCSR,R5 ;SET TABLE POINTER
1191 004036 160025 SUB RO,(R5)+
1192 004040 160025 SUB RO,(R5)+
1193 004042 160025 SUB RO,(R5)+
1194 004044 160025 SUB RO,(R5)+
1195 004046 060025 ADD RO,(R5)+
1196 004050 060025 ADD RO,(R5)+
1197 004052 060025 ADD RO,(R5)+
1198 004054 000137 001754 JMP @#RSTART ;RESTART TEST USING NEXT DR11-C
1199 004060 104406 CKSWR
1200 004062 032777 001000 175126 BIT #SW9,@SWR
1201 004070 001004 BNE BS
1202 004072 104401 5S: TYPE ;TYPE 'END PASS'
1203 004074 076675 MESS3
1204 004076 005267 175144 INC PASCNT ;KEEP TRACK OF PASSES COMPLETED
1205 004102 013700 000042 8S: MOV @#42,R0
1206 004106 001405 BEQ END1
1207 004110 000005 RESET
1208 004112 004710 $ENDAD: JSR PC,(R0)
1209 004114 000240 NOP
1210 004116 000240 NOP
1211 004120 000240 NOP
1212 004122 000137 001726 END1: JMP @#RSTART
1213
1214 ;ENTERED WITH SYSTEM TRAP CALL(HLT)
1215 ;PRINT OUT THE ERROR PC AND STATUS REGISTER
1216 004126 104406 .HLT: CKSWR
1217 004130 037727 175062 020000 BIT @SWR,@SW13 ;TEST FOR INHIBIT PRINT OUT
1218 004136 001401 BEQ .+4 ;BRANCH TO PRINT
1219 004140 000002 RTI ;INHIBIT RETURN TO MAIN STREAM
1220 004142 012667 000172 MOV (6)+,SAVPC ;PC OF FAILING ROUTINE
1221 004146 012667 000170 MOV (6)+,SAVCC ;CC OF ERROR CONDITION
1222 004152 024646 CMP -(6),-(6) ;REPOSITION THE STACK
1223 004154 105777 000156 TSTB @TCSR ;WAIT FOR FLAG
1224 004160 100375 BPL .-4 ;IF NOT UP.
1225 004162 012777 000215 000144 MOV #215,@TDBR ;CR
1226 004170 105777 000142 TSTB @TCSR
1227 004174 100375 BPL .-4
1228 004176 012777 000212 000130 MOV #212,@TDBR ;LINE FEED
1229 004204 105777 000126 TSTB @TCSR
1230 004210 100375 BPL .-4
1231 004212 010267 000110 MOV %2,SAVR2 ;SAVE R2
1232 004216 010367 000106 MOV %3,SAVR3 ;SAVE R3
1233 004222 010467 000104 MOV %4,SAVR4 ;SAVE R4
1234 004226 016702 000106 MOV SAVPC,%2
1235 004232 004767 000106 JSR %7,PRTAB ;PRINT OCTAL NUMBER
1236 004236 012777 000240 000070 MOV #240,@TDBR
1237 004244 105777 000066 TSTB @TCSR ;SPACE BETWEEN WORDS
1238 004250 100375 BPL .-4
1239 004252 016702 000064 MOV SAVCC,%2
1240 004256 004767 000062 JSR %7,PRTAB ;PRINT OCTAL NUMBER

```

1241	004262	004767	000522		JSR	%7, MORE10	; DEVICE ADDRESS AND VECTORS
1242	004266	016702	000034		MOV	SAVR2, %2	; RESTORE REGISTERS
1243	004272	016703	000032		MOV	SAVR3, %3	
1244	004276	016704	000030		MOV	SAVR4, %4	
1245	004302	023737	000042	000046	CMP	@#42, @#46	; ARE WE IN ACT-11 AUTO MODE?
1246	004310	001404			BEQ	1\$	; HALT ON ERROR IF YES
1247	004312	104406			CKSWR		
1248	004314	005777	174676		TST	@SWR	; TEST FOR HALT SWITCH
1249	004320	100001			BPL	.+4	
1250	004322	000000		1\$:	HALT		; HALT ON ERROR SET
1251	004324	000002			RTI		; RETURN TO MAIN STREAM
1252	004326	000000			SAVR2:	0	
1253	004330	000000			SAVR3:	0	
1254	004332	000000			SAVR4:	0	
1255	004334	177566			TDBR:	177566	; DATA
1256	004336	177564			TCSR:	177564	; STATUS
1257	004340	000000			SAVPC:	0	
1258	004342	000000			SAVCC:	0	
1259							
1260	004344	005067	000252		PRTAB:	CLR	BINCT
1261	004350	005067	000244		CLR	WGCT	
1262	004354	012704	004626		MOV	#LIST, %4	; GET LIST ADDRESS
1263	004360	012767	000005	000236	MOV	#5, ASCNT	
1264	004366	012767	000007	000220	MOV	#7, SEVEN	
1265	004374	012767	000001	000214	MOV	#1, DECML	
1266	004402	105777	177730		WAIT1:	TSTB	@TCSR
1267	004406	100375			BPL	WAIT1	
1268	004410	005702			TST	%2	
1269	004412	100404			BMI	MINUS	; NEG SIGN PRINT 1
1270	004414	012777	000260	177712	MOV	#260, @TDBR	; POS SIGN PRINT 0
1271	004422	000403			BR	STAR	
1272	004424	012777	000261	177702	MINUS:	MOV	#261, @TDBR
1273	004432	016703	000156		STAR:	MOV	SEVEN, %3
1274	004436	010267	000150		MOV	%2, TOODLE	; PUT MASK IN R3
1275	004442	005167	000144		COM	TOODLE	; GET READY TO DOODLE NUMBER IN TOODLE
1276	004446	046703	000140		BIC	TOODLE, %3	; COMPENSATES FOR COMPLEMENT DURING BIC
1277	004452	001410			BEQ	WRTOC	; AND IN OCTAL CHARACTER
1278	004454	066767	000136	000136	MKNUM:	ADD	DECML, WGCT
1279	004462	005267	000134		INC	BINCT	; COUNT UP TO
1280	004466	026703	000126		CMP	WGCT, %3	; AND RECORD
1281	004472	001370			BNE	MKNUM	; SAME BINARY WEIGHT
1282	004474	062767	000260	000120	WRTOC:	ADD	#260, BINCT
1283	004502	016724	000114		MOV	BINCT, (4)+	; KEEP COUNTN
1284	004506	066767	000102	000102	ADD	SEVEN, DECML	; ADD ASCII PREFIX
1285	004514	005067	000100		CLR	WGCT	; WRITE ASCII CHAR IN LIST
1286	004520	005067	000076		CLR	BINCT	; EXPAND BINARY WEIGHT
1287	004524	005367	000074		DEC	ASCNT	
1288	004530	001410			BEQ	XLIST	; 5 CHAR IN LIST
1289	004532	012703	000003		MOV	#3, %3	; SET X3 FOR ADD LOOP
1290	004536	066767	000052	000050	MOADD:	ADD	SEVEN, SEVEN
1291	004544	005303			DEC	%3	; MAKING SEVENTY BY SEVEN
1292	004546	001373			BNE	MOADD	
1293	004550	000730			BR	STAR	; NX SEVEN SET GET NX OCTAL
1294	004552	012767	000005	000044	XLIST:	MOV	#5, ASCNT
1295	004560	105777	177552		WAIT2:	TSTB	@TCSR
1296	004564	100375			BPL	WAIT2	; SEND 5 CHAR TO TTY

```

1297 004566 014477 177542      MOV      -(4),@TDBR
1298 004572 005367 000026      DEC      ASCNT
1299 004576 001401                BEQ      HDFHM          ;FINISH PRINTING GET NXT NUM
1300 004600 000767                BR       WAIT2
1301 004602 105777 177530      HDFHM:  TSTB @TCSR
1302 004606 100375                BPL     -4
1303 004610 000207                RTS     %7          ;HEAD FOR HOME
1304 004612 000000      TOODLE: 0
1305 004614 000000      SEVEN:  0
1306 004616 000000      DECML:  0
1307 004620 000000      WGTCT:  0
1308 004622 000000      BINCT:  0
1309 004624 000000      ASCNT:  0
1310 004626 000000      LIST:   0
1311 004630 000000                0
1312 004632 000000                0
1313 004634 000000                0
1314 004636 000000                0
1315                ;SCOPE LOOP ROUTINE ENTERED BY USER TRAP
1316 004640 022606      SCOPEB:  CMP      (6)+,%6          ;REPOSITION THE STACK
1317 004642 012637 177776      MOV      (6)+,@#PSW
1318 004646 000177 000120      JMP      @RETURN          ;SCOPE RETURN
1319
1320                ;SCOPE OR/AND ITERATION LOOP FOR EACH TEST 4000 TIMES
1321 004652 104406      .SCOPE:  CKSWR
1322 004654 032777 040000 174334      BIT      #SW14,@SWR          ;TEST SR FOR SCOPE
1323 004662 001366                BNE     SCOPEB          ;YES SCOPE
1324 004664 005737 001240      TST      @#XORFLG
1325 004670 100012                BPL     1$
1326 004672 013746 000004      MOV      @#4,-(%6)
1327 004676 012737 005000 000004      MOV      #XOR,@#4
1328 004704 012737 000031 177060      MOV      #31,@#177060
1329 004712 012637 000004      MOV      (%6)+,@#4
1330 004716 104406      1$:      CKSWR
1331 004720 032777 004000 174270      BIT      #SW11,@SWR          ;NO - TEST FOR ITERATION
1332 004726 001014                BNE     SCOPEA          ;INHIBIT ITERATION
1333 004730 005767 174312      TST      PASCNT          ;ARE WE IN FIRST PASS?
1334 004734 001411                BEQ     SCOPEA          ;BRANCH IF YES; INHIBIT ITERATIONS
1335 004736 026767 000026 000022      CMP      SCOPEF,ICOUNT
1336 004744 001403                BEQ     SCOPEF          ;EXIT - DONE
1337 004746 005267 000016      INC      SCOPEF          ;INCREMENT COUNT
1338 004752 000732                BR      SCOPEB          ;LOOP SOME MORE
1339 004754 005067 000010      SCOPEG:  CLR      SCOPEF          ;CLEAR COUNT
1340 004760 011667 000006      SCOPEA:  MOV      @%6,RETURN          ;SAVE SCOPE RETURN POINTER
1341 004764 000002                RTI
1342 004766 004000                ;RETURN INLINE-NEXT TEST
1343 004770 000000      ICOUNT:  4000
1344 004772 002010      SCOPEF:  0          ;COUNT LOCATION FOR ITERATION LOOP
1345 004774 000167 173200      RETURN:  BEGIN          ;ADDRESS OF LAST TEST
1346                JMP      200
1347 005000 022626      XOR:     CMP      (%6)+,(%6)+
1348 005002 012637 000004      MOV      (%6)+,@#4
1349 005006 000714                BR      SCOPEB
1350                ;PRINT DEVICE ADDRESS AND VECTOR
1351 005010 012777 000240 177316      MOREID:  MOV      #240,@TDBR
1352 005016 105777 177314      TSTB    @TCSR

```

```

1353 005022 100375      BPL      -4
1354 005024 013702 001222      MOV      @RCSR,%2
1355 005030 004767 177310      JSR      X7,PRTAB
1356 005034 012777 000240 177272      MOV      @240,@TDBR
1357 005042 105777 177270      TSTB    @TCSR
1358 005046 100375      BPL      -4
1359 005050 016702 174156      MOV      DRVECA,%2
1360 005054 004767 177264      JSR      X7,PRTAB
1361 005060 012777 000215 177246      MOV      @215,@TDBR
1362 005066 105777 177244      TSTB    @TCSR
1363 005072 100375      BPL      -4
1364 005074 012777 000212 177232      MOV      @212,@TDBR
1365 005102 105777 177230      TSTB    @TCSR
1366 005106 100375      BPL      -4
1367 005110 005077 177220      CLR      @TDBR
1368 005114 105777 177216      TSTB    @TCSR
1369 005120 100375      BPL      -4
1370 005122 000207      RTS      %7      ;BACK TO PRINT
1371
1372      ;ENTER HERE FOR POWER FAIL
1373
1374 005124 010046      PFAIL:  MOV      %0,-(6)      ;SAVE REGISTER OR STACK
1375 005126 010146      MOV      %1,-(6)      ;WHEN POWERING DOWN
1376 005130 010246      MOV      %2,-(6)
1377 005132 010346      MOV      %3,-(6)
1378 005134 010446      MOV      %4,-(6)
1379 005136 010546      MOV      %5,-(6)
1380 005140 016746 172660      MOV      24,-(6)
1381 005144 010637 005160      MOV      %6,@SAVR6      ;STORE STACK POSITION
1382 005150 012737 005162 000024      MOV      @RESTAR,@24
1383 005156 000000      HALT
1384 005160 000000      SAVR6:  0      ;HALT ON POWER DOWN NORMAL
1385 005162 016706 177772      RESTAR: MOV      SAVR6,%6      ;STACK IS SAVED HERE
1386 005166 012667 172632      MOV      (6)+,%4      ;RESTORE REGISTER OFF STACK
1387 005172 012605      MOV      (6)+,%5      ;WHEN POWERING UP
1388 005174 012604      MOV      (6)+,%4
1389 005176 012603      MOV      (6)+,%3
1390 005200 012602      MOV      (6)+,%2
1391 005202 012601      MOV      (6)+,%1
1392 005204 012600      MOV      (6)+,%0
1393 005206 000137 001754      JMP      @RSTART
1394
1395      ;ENTER HERE FOR UNIQUE SELECTION OF DR11C
1396
1397 005212 000000      SPEC0:  HALT      ;PLACE ADDRESS OF DR11-C CONTROL STATUS
1398 005214 104406      CKSWR
1399 005216 017701 173774      MOV      @SWR,R1
1400 005222 004737 005246      JSR      PC,@SPEC1
1401 005226 000000      HALT
1402 005230 104406      CKSWR
1403 005232 017701 173760      MOV      @SWR,R1
1404 005236 004737 005276      JSR      PC,@SPEC2
1405 005242 000137 001726      JMP      @START
1406
1407 005246 012700 001200      SPEC1:  MOV      @RCSR,R0      ;SET TABLE ADDRESS
1408 005252 010120      MOV      R1,(R0)+      ;LOAD INTO TABLE STARTING AT RCSR

```

```

1409 005254 062701 000002      ADD    #2,R1          ;STEP TO ADDRESS OF DROUTBUF
1410 005260 010120              MOV    R1,(R0)+       ;LOAD INTO TABLE
1411 005262 062701 000002      ADD    #2,R1          ;STEP TO ADDRESS OF DRINBUF
1412 005266 010120              MOV    R1,(R0)+       ;LOAD INTO TABLE
1413 005270 005301              DEC    R1             ;FORM ADDRESS OF DROUTBUF+1
1414 005272 010120              MOV    R1,(R0)+       ;LOAD INTO TABLE
1415 005274 000207              RTS    PC
1416
1417 005276 012700 001210      SPEC2: MOV    #RCSR1,R0
1418 005302 010120              MOV    R1,(R0)+       ;LOAD INTO TABLE
1419 005304 005721              TST    (R1)+
1420 005306 010120              MOV    R1,(R0)+
1421 005310 005721              TST    (R1)+
1422 005312 010120              MOV    R1,(R0)+
1423 005314 000207              RTS    PC
1424
1425
1426 005316
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444 005316 105767 000223      $TYPE: TSTB   $TPFLG    ;IS THERE A TERMINAL?
1445 005322 100002              BPL    1$             ;BR IF YES
1446 005324 000000              HALT                   ;HALT HERE IF NO TERMINAL
1447 005326 000407              BR     3$             ;LEAVE
1448 005330 010046              1$:  MOV    RO,-(SP)    ;SAVE RO
1449 005332 017600 000002      MOV    #2(SP),RO      ;GET ADDRESS OF ASCIZ STRING
1450 005336 112046              2$:  MOVB   (RO)+,-(SP) ;PUSH CHARACTER TO BE TYPED ONTO STACK
1451 005340 001005              BNE    4$             ;BR IF IT ISN'T THE TERMINATOR
1452 005342 005726              TST    (SP)+          ;IF TERMINATOR POP IT OFF THE STACK
1453 005344 012600              60$: MOV    (SP)+,RO    ;RESTORE RO
1454 005346 062716 000002      3$:  ADD    #2,(SP)     ;ADJUST RETURN PC
1455 005352 000002              RTI                    ;RETURN
1456 005354 122716 000011      4$:  CMPB   #HT,(SP)    ;BRANCH IF <HT>
1457 005360 001430              BEQ    8$             ;BRANCH IF NOT <CRLF>
1458 005362 122716 000200      CMPB   #CRLF,(SP)
1459 005366 001006              BNE    5$             ;POP <CR><LF> EQUIV
1460 005370 005726              TST    (SP)+          ;TYPE A CR AND LF
1461 005372 104401              TYPE
1462 005374 005547              $CRLF
1463 005376 105067 000130      CLRB   $CHARCNT      ;CLEAR CHARACTER COUNT
1464 005402 000755              BR     2$             ;GET NEXT CHARACTER

```

```

.LIST ME
TYP:
.SBTTL TYPE ROUTINE

```

```

*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: SNUL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLS CO TAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR ;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
* TYPE
* MESADR
*

```

```

TYPE ROUTINE
1465 005404 004767 000056 5$: JSR PC,$TYPEC ;:GO TYPE THIS CHARACTER
1466 005410 126726 000130 6$: CMPB $FILLC,(SP)+ ;:IS IT TIME FOR FILLER CHARS.?
1467 005414 001350 BNE 2$ ;:IF NO GO GET NEXT CHAR.
1468 005416 016746 000120 MOV $NULL,-(SP) ;:GET # OF FILLER CHARS. NEEDED
1469 ;:AND THE NULL CHAR.
1470 005422 105366 000001 7$: DECIB ;:(SP) ;:DOES A NULL NEED TO BE TYPED?
1471 005426 002770 BLT 6$ ;:BR IF NO--GO POP THE NULL OFF OF STACK
1472 005430 004767 000032 JSR PC,$TYPEC ;:GO TYPE A NULL
1473 005434 105367 000072 DECIB $CHARCNT ;:DO NOT COUNT AS A COUNT
1474 005440 000770 BR 7$ ;:LOOP
1475
1476 ;:HORIZONTAL TAB PROCESSOR
1477
1478 005442 112716 000040 8$: MOVB #' (SP) ;:REPLACE TAB WITH SPACE
1479 005446 004767 000014 9$: JSR PC,$TYPEC ;:TYPE A SPACE
1480 005452 132767 000007 000052 BITB #',$CHARCNT ;:BRANCH IF NOT AT
1481 005460 001372 BNE 9$ ;:TAB STOP
1482 005462 005726 TST (SP)+ ;:POP SPACE OFF STACK
1483 005464 000724 BR 2$ ;:GET NEXT CHARACTER
1484 005466 105777 000044 $TYPEC: TSTB 2$STPS ;:WAIT UNTIL PRINTER IS READY
1485 005472 100375 BPL $TYPEC
1486 005474 116677 000002 000036 MOVB 2(SP),2$STPB ;:LOAD CHAR TO BE TYPED INTO DATA REG.
1487 005502 122766 000015 000002 CMPB #CR,2(SP) ;:IS CHARACTER A CARRIAGE RETURN?
1488 005510 001003 BNE 1$ ;:BRANCH IF NO
1489 005512 105067 000014 CLRIB $CHARCNT ;:YES--CLEAR CHARACTER COUNT
1490 005516 000406 BR $TYPEX ;:EXIT
1491 005520 122766 000012 000002 1$: CMPB #LF,2(SP) ;:IS CHARACTER A LINE FEED?
1492 005526 001402 BEQ $TYPEX ;:BRANCH IF YES
1493 005530 105227 INCB (PC)+ ;:COUNT THE CHARACTER
1494 005532 000000 $CHARCNT: .WORD 0 ;:CHARACTER COUNT STORAGE
1495 005534 000207 $TYPEX: RTS PC
1496
1497 005536 177564 $STPS: .WORD 177564 ;:TTY PRINTER STATUS REG. ADDRESS
1498 005540 177566 $STPB: .WORD 177566 ;:TTY PRINTER BUFFER REG. ADDRESS
1499 005542 000 $NULL: .BYTE 0 ;:CONTAINS NULL CHARACTER FOR FILLS
1500 005543 002 $FILLS: .BYTE 2 ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
1501 005544 012 $FILLC: .BYTE 12 ;:INSERT FILL CHARS. AFTER A "LINE FEED"
1502 005545 000 $STPLG: .BYTE 0 ;:"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1503 005546 077 $QUES: .ASCII "? " ;:QUESTION MARK
1504 005547 015 $CRLF: .ASCII <15> ;:CARRIAGE RETURN
1505 005550 000012 $LF: .ASCIZ <12> ;:LINEFEED
1506 ;:SBTTL BINARY TO OCTAL (ASCII) AND TYPE
1507
1508 ;:*****
1509 ;:THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1510 ;:OCTAL (ASCII) NUMBER AND TYPE IT.
1511 ;:$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1512 ;:$CALL:
1513 ;: MOV NUM,-(SP) ;:NUMBER TO BE TYPED
1514 ;: TYPOS ;:CALL FOR TYPEOUT
1515 ;: .BYTE N ;:N=! TO 6 FOR NUMBER OF DIGITS TO TYPE
1516 ;: .BYTE M ;:M=1 OR 0
1517 ;: ;:1=TYPE LEADING ZEROS
1518 ;: ;:0=SUPPRESS LEADING ZEROS
1519 ;:
1520 ;:$STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
    
```

```

1521 ;*STYPOS OR STYPOC
1522 ;*CALL:
1523 ;*      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
1524 ;*      TYPON      ;; CALL FOR TYPEOUT
1525 ;*
1526 ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1527 ;*CALL:
1528 ;*      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
1529 ;*      TYPOC      ;; CALL FOR TYPEOUT
1530
1531 005552 017646 000000      STYPOS: MOV      2(SP),-(SP)      ;; PICKUP THE MODE
1532 005556 116667 000001 000211  MOVB     1(SP),SOFILL      ;; LOAD ZERO FILL SWITCH
1533 005564 112667 000207      MOVB     (SP)+,SOMODE+1    ;; NUMBER OF DIGITS TO TYPE
1534 005570 062716 000002      ADD      #2,(SP)          ;; ADJUST RETURN ADDRESS
1535 005574 000406      BR      STYPON
1536 005576 112767 000001 000171  STYPOC: MOVB     #1,SOFILL      ;; SET THE ZERO FILL SWITCH
1537 005604 112767 000006 000165  MOVB     #6,SOMODE+1      ;; SET FOR SIX(6) DIGITS
1538 005612 112767 000005 000154  STYPON: MOVB     #5,SOCNT      ;; SET THE ITERATION COUNT
1539 005620 010346      MOV      R3,-(SP)        ;; SAVE R3
1540 005622 010446      MOV      R4,-(SP)        ;; SAVE R4
1541 005624 010546      MOV      R5,-(SP)        ;; SAVE R5
1542 005626 116704 000145      MOVB     SOMODE+1,R4      ;; GET THE NUMBER OF DIGITS TO TYPE
1543 005632 005404      NEG      R4
1544 005634 062704 000006      ADD      #6,R4            ;; SUBTRACT IT FOR MAX. ALLOWED
1545 005640 110467 000132      MOVB     R4,SOMODE        ;; SAVE IT FOR USE
1546 005644 116704 000125      MOVB     SOFILL,R4        ;; GET THE ZERO FILL SWITCH
1547 005650 016605 000012      MOV      12(SP),R5       ;; PICKUP THE INPUT NUMBER
1548 005654 005003      CLR      R3              ;; CLEAR THE OUTPUT WORD
1549 005656 006105      15:     ROL      R5        ;; ROTATE MSB INTO "C"
1550 005660 000404      BR      35              ;; GO DO MSB
1551 005662 006105      25:     ROL      R5        ;; FORM THIS DIGIT
1552 005664 006105      ROL      R5
1553 005666 006105      ROL      R5
1554 005670 010503      MOV      R5,R3
1555 005672 006103      35:     ROL      R3        ;; GET LSB OF THIS DIGIT
1556 005674 105367 000076      DECB     SOMODE          ;; TYPE THIS DIGIT?
1557 005700 100016      BPL      75              ;; BR IF NO
1558 005702 042703 177770      BIC      #177770,R3      ;; GET RID OF JUNK
1559 005706 001002      BNE      45              ;; TEST FOR 0
1560 005710 005704      TST      R4              ;; SUPPRESS THIS 0?
1561 005712 001403      BEG      55              ;; BR IF YES
1562 005714 005204      45:     INC      R4        ;; DON'T SUPPRESS ANYMORE 0'S
1563 005716 052703 000060      BIS      #'0,R3          ;; MAKE THIS DIGIT ASCII
1564 005722 052703 000040      55:     BIS      #' ,R3    ;; MAKE ASCII IF NOT ALREADY
1565 005726 110367 000040      MOVB     R3,#5           ;; SAVE FOR TYPING
1566 005732 104401 005772      TYPE     #5              ;; GO TYPE THIS DIGIT
1567 005736 105367 000032      75:     DECB     SOCNT      ;; COUNT BY 1
1568 005742 003347      BGT      25              ;; BR IF MORE TO DO
1569 005744 002402      BLT      65              ;; BR IF DONE
1570 005746 005204      INC      R4              ;; INSURE LAST DIGIT ISN'T A BLANK
1571 005750 000744      BR      25              ;; GO DO THE LAST DIGIT
1572 005752 012605      65:     MOV      (SP)+,R5    ;; RESTORE R5
1573 005754 012604      MOV      (SP)+,R4        ;; RESTORE R4
1574 005756 012603      MOV      (SP)+,R3        ;; RESTORE R3
1575 005760 016666 000002 000004  MOV      2(SP),4(SP)     ;; SET THE STACK FOR RETURNING
1576 005766 012616      MOV      (SP)+,(SP)

```

1577 005770 000002  
1578 005772 000  
1579 005773 000  
1580 005774 000  
1581 005775 000  
1582 005776 000000

RTI ;: RETURN  
BS: .BYTE 0 ;: STORAGE FOR ASCII DIGIT  
 .BYTE 0 ;: TERMINATOR FOR TYPE ROUTINE  
SOCNT: .BYTE 0 ;: OCTAL DIGIT COUNTER  
SOFILL: .BYTE 0 ;: ZERO FILL SWITCH  
SOMODE: .WORD 0 ;: NUMBER OF DIGITS TO TYPE  
.SBTTL TTY INPUT ROUTINE

1586 006000 177560  
1587 006002 177562

\*\*\*\*\*  
\$TKS: .WORD 177560 ;: TTY KBD STATUS  
\$TKB: .WORD 177562 ;: TTY KBD BUFFER  
.ENABL LSB

1595 006004 022767 000176 173204  
1596 006012 001074  
1597 006014 105777 177760  
1598 006020 100071  
1599 006022 117746 177754  
1600 006026 042716 177600  
1601 006032 022726 000007  
1602 006036 001062  
1603 006040 126727 000514 000001  
1604 006046 001456

\*\*\*\*\*  
\*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.  
\*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL  
\*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL  
\*WHEN OPERATING IN TTY FLAG MODE.

1606 006050 104401 006531  
1607 006054 104401 006536  
1608 006060 016746 172112  
1609 006064 104402  
1610 006066 104401 006547  
1611 006072 005046  
1612 006074 005046  
1613 006076 105777 177676  
1614 006102 100375  
1615  
1616 006104 117746 177672  
1617 006110 042716 177600  
1618  
1619  
1620

\$CKSWR: CMP #SWREG,SWR ;: IS THE SOFT-SWR SELECTED?  
 BNE 15\$ ;: BRANCH IF NO  
 TSTB 2\$TKS ;: CHAR THERE?  
 BPL 15\$ ;: IF NO, DON'T WAIT AROUND  
 MOVB 2\$TKB,-(SP) ;: SAVE THE CHAR  
 BIC #177,(SP) ;: STRIP-OFF THE ASCII  
 CMP #7,(SP)+ ;: IS IT A CONTROL G?  
 BNE 15\$ ;: NO, RETURN TO USER  
 CMPB \$AUTOB,#1 ;: ARE WE RUNNING IN AUTO-MODE?  
 BEQ 15\$ ;: BRANCH IF YES

1621 006114 021627 000025  
1622 006120 001005  
1623 006122 104401 006524  
1624 006126 062706 000006  
1625 006132 000757  
1626  
1627  
1628 006134 021627 000015  
1629 006140 001022  
1630 006142 005766 000004  
1631 006146 001403  
1632 006150 016677 000002 173040

\$GTSWR: TYPE ,SCNTLG ;: ECHO THE CONTROL-G (+G)  
 TYPE \$TKSWR ;: TYPE CURRENT CONTENTS  
 MOV SWREG,-(SP) ;: SAVE SWREG FOR TYPEOUT  
 TYOC ;: GO TYPE--OCTAL ASCII(ALL DIGITS)  
 TYPE ,SMNEW ;: PROMPT FOR NEW SWR  
 19\$: CLR -(SP) ;: CLEAR COUNTER  
 CLR -(SP) ;: THE NEW SWR  
 7\$: TSTB 2\$TKS ;: CHAR THERE?  
 BPL 7\$ ;: IF NOT TRY AGAIN

1621 006114 021627 000025  
1622 006120 001005  
1623 006122 104401 006524  
1624 006126 062706 000006  
1625 006132 000757  
1626  
1627  
1628 006134 021627 000015  
1629 006140 001022  
1630 006142 005766 000004  
1631 006146 001403  
1632 006150 016677 000002 173040

9\$: CMP (SP),#25 ;: IS IT A CONTROL-U?  
 BNE 10\$ ;: BRANCH IF NOT  
 TYPE ,SCNTLU ;: YES, ECHO CONTROL-U (+U)  
 20\$: ADD #6,SP ;: IGNORE PREVIOUS INPUT  
 BR 19\$ ;: LET'S TRY IT AGAIN

1628 006134 021627 000015  
1629 006140 001022  
1630 006142 005766 000004  
1631 006146 001403  
1632 006150 016677 000002 173040

10\$: CMP (SP),#15 ;: IS IT A <CR>?  
 BNE 16\$ ;: BRANCH IF NO  
 TST 4(SP) ;: YES, IS IT THE FIRST CHAR?  
 BEQ 11\$ ;: BRANCH IF YES  
 MOV 2(SP),2\$SWR ;: SAVE NEW SWR



```

1633 006156 062706 000006
1634 006162 104401 005547
1635 006168 126727 000367 000001
1636 006174 001003
1637 006176 012777 000100 177574
1638 006204 000002
1639 006206 004767 177254
1640 006212 021627 000060
1641 006216 002420
1642 006220 021627 000067
1643 006224 003015
1644 006226 042726 000060
1645 006232 005766 000002
1646 006236 001403
1647 006240 006316
1648 006242 006316
1649 006244 006316
1650 006246 005266 000002
1651 006252 056616 177776
1652 006256 000707
1653 006260 104401 005546
1654 006264 000720
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666 006266 011646
1667 006270 016666 000004 000002
1668 006276 105777 177476
1669 006302 100375
1670 006304 117766 177472 000004
1671 006312 042766 177600 000004
1672 006320 026627 000004 000023
1673 006326 001013
1674 006330 105777 177444
1675 006334 100375
1676 006336 117746 177440
1677 006342 042716 177600
1678 006346 022627 000021
1679 006352 001366
1680 006354 000750
1681 006356 026627 000004 000140
1682 006364 002407
1683 006366 026627 000004 000175
1684 006374 003003
1685 006376 042766 000040 000004
1686 006404 000002
1687
1688

```

```

11$: ADD #6,SP ;: CLEAR UP STACK
14$: TYPE $CRLF ;: ECHO <CR> AND <LF>
CMPB $INTAG,#1 ;: RE-ENABLE TTY KBD INTERRUPTS?
BNE 15$ ;: BRANCH IF NOT
MOV #100,@$TKS ;: RE-ENABLE TTY KBD INTERRUPTS
15$: RTI ;: RETURN
16$: JSR PC,$TYPEC ;: ECHO CHAR
CMP (SP),#60 ;: CHAR < 0?
BLT 18$ ;: BRANCH IF YES
CMP (SP),#67 ;: CHAR > 7?
BGT 18$ ;: BRANCH IF YES
BIC #60,(SP)+ ;: STRIP-OFF ASCII
TST 2(SP) ;: IS THIS THE FIRST CHAR
BEQ 17$ ;: BRANCH IF YES
ASL (SP) ;: NO, SHIFT PRESENT
ASL (SP) ;: CHAR OVER TO MAKE
ASL (SP) ;: ROOM FOR NEW ONE.
17$: INC 2(SP) ;: KEEP COUNT OF CHAR
BIS -2(SP),(SP) ;: SET IN NEW CHAR
BR 7$ ;: GET THE NEXT ONE
18$: TYPE $QUES ;: TYPE ?<CR><LF>
BR 20$ ;: SIMULATE CONTROL-U
.DSABL LSB

```

```

;:*****
;: 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 @$TKS ;: WAIT FOR
BPL 1$ ;: A CHARACTER
MOV 4(SP),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$: TSTB @$TKS ;: WAIT FOR A CHARACTER
BPL 2$ ;: LOOP UNTIL ITS THERE
MOV 4(SP),-(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

```

```

1689 ;*CALL:
1690 ;* RDLIN ;: INPUT A STRING FROM THE TTY
1691 ;* RETURN HERE ;: ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
1692 ;* ;: TERMINATOR WILL BE A BYTE OF ALL 0'S
1693
1694 006406 010346 $RDLIN: MOV R3, -(SP) ;: SAVE R3
1695 006410 012703 006514 1$: MOV #STTYIN, R3 ;: GET ADDRESS
1696 006414 022703 006524 2$: CMP #STTYIN+8., R3 ;: BUFFER FULL?
1697 006420 101405 4$: BLOS 4$ ;: BR IF YES
1698 006422 104407 ;: GO READ ONE CHARACTER FROM THE TTY
1699 006424 112613 ;: GET CHARACTER
1700 006426 122713 000177 10$: CMPB #177, (R3) ;: IS IT A RUBOUT
1701 006432 001003 ;: SKIP IF NOT
1702 006434 104401 005546 4$: TYPE $QUES ;: TYPE A '?'
1703 006440 000763 ;: CLEAR THE BUFFER AND LOOP
1704 006442 111367 000044 3$: MOVB (R3), 9$ ;: ECHO THE CHARACTER
1705 006446 104401 006512 ;:
1706 006452 122723 000015 ;: CHECK FOR RETURN
1707 006456 001356 ;: LOOP IF NOT RETURN
1708 006460 105063 177777 ;: CLEAR RETURN (THE 15)
1709 006464 104401 005550 ;: TYPE A LINE FEED
1710 006470 012603 ;: RESTORE R3
1711 006472 011646 ;: ADJUST THE STACK AND PUT ADDRESS OF THE
1712 006474 016666 000004 000002 ;: FIRST ASCII CHARACTER ON IT
1713 006502 012766 006514 000004 ;:
1714 006510 000002 ;: RETURN
1715 006512 000 ;: STORAGE FOR ASCII CHAR. TO TYPE
1716 006513 000 ;: TERMINATOR
1717 006514 000010 ;: RESERVE 8 BYTES FOR TTY INPUT
1718 006524 052536 005015 000 $TTYIN: .BLKB 8. ;: CONTROL "U"
1719 006531 136 006507 000012 $CNTLU: .ASCIZ /+U/<15><12> ;: CONTROL "G"
1720 006536 005015 053523 020122 $CNTLG: .ASCIZ /+G/<15><12>
1721 006544 020075 000 $MSWR: .ASCIZ <15><12>/SWR = /
1722 006547 040 047040 053505 $MNEW: .ASCIZ / NEW = /
1723 006554 036440 000040 ;:
1724 006560 000 ;:
1725 006561 000 ;:
1726 ;:
1727 ;:
1728 ;: *****
1729 ;: *THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1730 ;: *AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1731 ;: *OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1732 ;: *GO TO THAT ROUTINE.
1733 ;:
1734 006562 010046 $TRAP: MOV R0, -(SP) ;: SAVE R0
1735 006564 016600 000002 ;: GET TRAP ADDRESS
1736 006570 005740 ;: BACKUP BY 2
1737 006572 111000 ;: GET RIGHT BYTE OF TRAP
1738 006574 006300 ;: POSITION FOR INDEXING
1739 006576 016000 006616 ;: INDEX TO TABLE
1740 006602 000200 ;: GO TO ROUTINE
1741 ;:
1742 ;:
1743 ;:
1744 ;: THIS IS USE TO HANDLE THE "GETPRI" MACRO

```

```

1745 006604 011646 $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
1746 006606 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
1747 006614 000002 RTI ;;RESTORE THE PSW
1748
1749 .SBTTL TRAP TABLE
1750
1751 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1752 ;*BY THE "TRAP" INSTRUCTION.
1753
1754 ; ROUTINE
1755 ; -----
1756 006616 006604 $TRPAD: .WORD $TRAP2
1757 006620 005316 $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
1758 006622 005576 $TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
1759 006624 005552 $TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
1760 006626 005612 $TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
1761
1762 006630 006054 $GTSWR ;;CALL=GTSWR TRAP+5(104405) GET SOFT-SWR SETTING
1763
1764 006632 006004 $CKSWR ;;CALL=CKSWR TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
1765 006634 006266 $RDCHR ;;CALL=RDCHR TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
1766 006636 006406 $RDLIN ;;CALL=RDLIN TRAP+10(104410) TTY TYPEIN STRING ROUTINE
1767 006640 005015 047531 020125 MESS1: .ASCIZ <15><12>'YOU ARE ON AN XOR TESTER'<15><12>
1768 006646 051101 020105 047117
1769 006654 040440 020116 047530
1770 006662 020122 042524 052123
1771 006670 051105 005015 000
1772 006675 105 042116 050040 MESS3: .ASCIZ /END PASS/<15><12>
1773 006702 051501 006523 000012
1774 000001 .END
    
```







TRTVEC=	000014	422#												
TST4	003132	1058	1063#											
TST5	003202	1070	1075#											
TST6	002760	1027#	1035											
TST9	003014	1033	1036#											
TYP	005316	696	1426#											
TYPE =	104401	764	821	1202	1461	1566	1606	1607	1610	1623	1634	1653	1702	1705
		1709	1757#											
TYPOC =	104402	1609	1758#											
TYPON =	104404	1760#												
TYPOS =	104403	1759#												
WAIT1	004402	1266#	1267											
WAIT2	004560	1295#	1296	1300										
WGTCT	004620	1261#	1278#	1280	1285*	1307#								
WRTOC	004474	1277	1282#											
XLIST	004552	1288	1294#											
XOR	005000	1327	1347#											
XORA	001712	813	824#											
XORFLG	001240	733#	816#	826*	1324									
\$AUTOB	006560	772*	1603	1724#										
\$CHARC	005532	1463*	1473*	1480	1489*	1494#								
\$CKSWR	006004	1595#	1764											
\$CNTAG=	***** U	741												
\$CNTLG	006531	1606	1719#											
\$CNTLU	006524	1623	1718#											
\$CRLF	005547	1462	1504#	1634	1718									
\$ENDAD	004112	699	1208#											
\$FILLC	005544	1466	1501#											
\$FILLS	005543	1500#												
\$GTSWR	006054	1607#	1762											
\$INTAG	006561	1635	1725#											
\$LF	005550	1505#	1709	1718										
\$MAIL =	***** U	758	768	1450										
\$MNEW	006547	1610	1722#											
\$MSWR	006536	1607	1720#											
\$MULL	005542	1468	1499#											
\$OCNT	005774	1538*	1567*	1580#										
\$OMODE	005776	1533*	1537*	1542	1545*	1556*	1582#							
\$QUES	005546	1503#	1653	1702	1718									
\$ROCHR	006266	1666#	1765											
\$RODEC=	***** U	1767												
\$ROLIN	006406	1694#	1766											
\$ROOCT=	***** U	1767												
\$ROSZ =	000010	1687#												
\$R2A =	***** U	1767												
\$SAVRE=	***** U	1767												
\$SETUP=	000100	431#	742	764	765	1590	1724							
\$STUP =	177777	431#												
\$TKB	006002	1587#	1599	1616	1670	1676								
?TKS	006000	1586#	1597	1613	1637*	1668	1674							
\$TPB	005540	1486#	1498#											
\$TIFLG	005545	1444	1502#											
\$TPS	005536	1484	1497#											
\$TRAF	006562	1734#												
\$TRAF2	006604	1745#	1756											
\$TRP =	000011	1773#	1758#	1759#	1760#	1761#	1762	1763#	1764	1765#	1766#	1767#		

\$TRPAD	006616	1739	1756#																				
\$TTYIN	006514	1695	1696	1713	1717#																		
\$TYPEB=	***** U	1761																					
\$TYPOS=	***** U	1761																					
\$TYPE	005316	1444#	1749	1757																			
\$TYPEC	005466	1465	1472	1479	1484#	1485	1639																
\$TYPEX	005534	1490	1492	1495#																			
\$TYPOC	005576	1536#	1758																				
\$TYPON	005612	1535	1538#	1760																			
\$TYPOS	005552	1531#	1759																				
\$OFILL	005775	1532*	1536*	1546	1581#																		
=	006710	432#	433	435	437	439	441	443	445	447	449	451	453	455									
		457	459	461	463	465	467	469	471	473	475	477	479	481									
		483	485	487	489	491	493	495	497	499	501	503	505	507									
		509	511	513	515	517	519	521	523	525	527	529	531	533									
		535	537	539	541	543	545	547	549	551	553	555	557	559									
		561	563	565	567	569	571	573	575	577	579	581	583	585									
		587	589	591	593	595	597	599	601	603	605	607	609	611									
		613	615	617	619	621	623	625	627	629	631	633	635	637									
		639	641	643	645	647	649	651	653	655	657	659	661	663									
		665	667	669	671	673	675	677	679	681	683	685	687	689									
		698#	700#	702#	705#	709#	866	874	882	889	904	918	926	937									
		947	952	960	966	973	981	985	993	1001	1007	1014	1021	1030									
		1043	1051	1141	1154	1167	1218	1224	1227	1230	1238	1249	1302	1353									
		1358	1363	1366	1369	1497	1498	1499	1500	1501	1502	1503	1504	1505									
		1506	1586	1587	1717#	1718	1724	1725	1726														
.HLT	004126	694	1216#																				
.SCOPE	004652	690	1321#																				

. ABS. 006710 000

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

DZDRCG, DZDRCG=SYSMAC.SML, DZDRCG.P11  
RUN-TIME: 139.4 SECONDS  
RUN-TIME RATIO: 159/23=6.7  
CORE USED: 32K (64 PAGES)