

PDP11/70

POWER FAIL TEST
MD-11-DEKBG-A

EP-DEKBG-A-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

This microfiche card contains a grid of frames. The frames on the left side contain data, likely test results or system status, organized in columns. The frames on the right side contain diagrams, possibly circuit board layouts or component connections. The text within the frames is too small to read clearly but appears to include various alphanumeric codes and symbols.

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DEKBG-A-D
PRODUCT NAME: PDP-11/70 POWER FAIL TEST
DATE CREATED: 1-MAR-75
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHORS: JIM LACEY

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 manual.

The software described in this document is furnished to the 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.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

Copyright (C) 1975 by Digital Equipment Corporation

CONTENTS

- 1. ABSTRACT
- 2. REQUIREMENTS
 - 2.1 Equipment
 - 2.2 Storage
 - 2.3 Preliminary programs
- 3. LOADING PROCEDURE
- 4. STARTING PROCEDURE
 - 4.1 Control switch settings
 - 4.2 Starting address
 - 4.3 Program and/or operator action
- 5. OPERATING PROCEDURE
 - 5.1 Operational switch settings
 - 5.2 Subroutine abstract
- 6. ERRORS
- 7. RESTRICTIONS
- 8. MISCELLANEOUS
 - 8.1 Execution time
 - 8.2 Stack pointer

1. Abstract

This program is made up of 16 subtests to check out the power fail on the 11/70. The 2 msec. power down and power up time is checked on each power fail. Initially power fails are tried in all processor modes then error conditions like red zone, yellow zone, time out, and odd address in all the processor modes. Finally a power fail is done with memory management aborts occurring and a memory volatility test is run on all available memory.

2. Requirements

2.1 Equipment

PDP11/70 standard computer with up to 1M words of core memory.

2.2 Storage

Program Storage - the routines use memory 0 - 4472

2.3 Preliminary programs

All processor diagnostics

3. Loading procedure

Use standard procedure for "XXDP" media.

4. Starting Procedure

4.1 Control switch settings

See 5.1.1

4.2 Starting address

Load Address 200 and Start.

5. Operating procedure

Load Address 200 and START. A message will be typed which is the name of the program, the size of memory, and running instructions. Turn the DISPLAY switch to DISPLAY REGISTER

and power down then up when the test number appears in the lights. Do this for each test until the count recycles to 1. Each subtest is executed once except test 16 which runs 8 times before continuing. SW14 causes the program loop on the current test..blank 2

5.1 Operational switch settings

At SA 200 ... all switches down will run through each test and HALT on error. SW14 should be used to loop on the current test.

5.1.1 Switch settings are:

sw<14> = 1 scope loop

5.2 Subroutine Abstracts

5.2.1 POWDOWN and POWUP

These routines are used to save and restore vital registers and test the time allowed for power fail by the processor. A SOB loop is used to check the timing. LOC 1100 contains the timing factor. Control is returned to the program via JMP (R3) so the power fail return address is put in R3. ILLUP and ILLDOWN are used for reporting not enough time to power down and up.

6. Errors

6.1 Error printout

None

6.2 Error HALTs

The program will HALT on error. The DISPLAY switch should be turned to the DATA PATHS position for the failing data. RO, which is displayed on a HALT, contains the bad data or bad address (see listing) in most of the tests.

If an error occurs in test 16, the data can be examined by turning the MODE switch to KERNAL I, load address with the address in RO and examine. To calculate the failing address, examine KIPARS (17772352) and use that for the offset to the address in RO. To do this, use KIPARS as bits <21:6> and add RO<12:0> to it. This is the physical address of the bad data.

If the processor HALT's at ILLUP, the power down routine did

not have enough time to complete. If it HALT's at ILLOWN, the processor powered down before the up routine completed. In both cases, 2 msec is the minimum time allowed by the processor. The program must be restarted at 200 after these errors. The address of the power failed routine is in location ERROR.

6.3 Error recovery
Hit continue or Restart at 200

7. Restrictions
None

8. Miscellaneous

8.1 Execution time
N/A

8.2 Stack Pointer
Stack is initially set to 1100

%

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

```
.TITLE MAINDEC-11-DEKBG-A      PDP-11/70 POWER FAIL
;*COPYRIGHT (C) 1975
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY JIM LACEY
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-A3).
;*
```

```
.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;* SWITCH USE
;* -----
;* 14 LOOP ON TEST
;* 10 INHIBIT BELL AT END-OF-PASS
```

```
.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100 ;;FIRST ADDRESS OF THE STACK
KERSTK= STACK ;;KERNEL STACK
SUPSTK= STACK-200 ;;SUPERVISOR STACK
```

001100
001100
000700

255
256
257
258

000600

177776

USESTK= STACK-300
.EQUIV EMT,ERROR
.EQUIV IOT,SCOPE
PS= 177776

CALL
:USER STACK
:BASIC DEFINITION OF ERROR CALL
:BASIC DEFINITION OF SCOPE CALL
:PROCESSOR STATUS WORD

```

269 .EQUIV PS,PSW
270 177774 STKLM= 177774 ;;STACK LIMIT REGISTER
271 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
272 177570 SWR= 177570 ;;SWITCH REGISTER
273 177570 DISPLAY=SWR
274
275 ;*MISCELLANEOUS DEFINITIONS
276 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
277 00012 LF= 12 ;;CODE LINE FEED
278 000015 CR= 15 ;;CODE CARRIAGE RETURN
279 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
280
281 ;*GENERAL PURPOSE REGISTER DEFINITIONS
282 000000 R0= %0 ;;GENERAL REGISTER
283 000001 R1= %1 ;;GENERAL REGISTER
284 000002 R2= %2 ;;GENERAL REGISTER
285 000003 R3= %3 ;;GENERAL REGISTER
286 000004 R4= %4 ;;GENERAL REGISTER
287 000005 R5= %5 ;;GENERAL REGISTER
288 000006 R6= %6 ;;GENERAL REGISTER
289 000007 R7= %7 ;;GENERAL REGISTER
290 .EQUIV R0,R10 ;;GENERAL REGISTER
291 .EQUIV R1,R11 ;;GENERAL REGISTER
292 .EQUIV R2,R12 ;;GENERAL REGISTER
293 .EQUIV R3,R13 ;;GENERAL REGISTER
294 .EQUIV R4,R14 ;;GENERAL REGISTER
295 .EQUIV R5,R15 ;;GENERAL REGISTER
296 .EQUIV R6,SP ;;STACK POINTER
297 .EQUIV SP,KSP ;;KERNEL STACK POINTER
298 .EQUIV SP,SSP ;;SUPERVISOR STACK POINTER
299 .EQUIV SP,USP ;;USER STACK POINTER
300 .EQUIV R7,PC ;;PROGRAM COUNTER
301
302 ;*PRIORITY LEVEL DEFINITIONS
303 000000 PR0= 0 ;;PRIORITY LEVEL 0
304 000040 PR1= 40 ;;PRIORITY LEVEL 1
305 000100 PR2= 100 ;;PRIORITY LEVEL 2
306 000140 PR3= 140 ;;PRIORITY LEVEL 3
307 000200 PR4= 200 ;;PRIORITY LEVEL 4
308 000240 PR5= 240 ;;PRIORITY LEVEL 5
309 000300 PR6= 300 ;;PRIORITY LEVEL 6
310 000340 PR7= 340 ;;PRIORITY LEVEL 7
311
312 ;*SWITCH REGISTER SWITCH DEFINITIONS
313 100000 SW15= 100000
314 040000 SW14= 40000
315 020000 SW13= 20000
316 010000 SW12= 10000
317 004000 SW11= 4000
318 002000 SW10= 2000
319 001000 SW09= 1000
320 000400 SW08= 400
321 000200 SW07= 200
322 000100 SW06= 100
323 000040 SW05= 40
324 000020 SW04= 20

```



```

325 000010 SW03= 10
326 000004 SW02= 4
327 000002 SW01= 2
328 000001 SW00= 1
329 .EQUIV SW09, SW9
330 .EQUIV SW08, SW8
331 .EQUIV SW07, SW7
332 .EQUIV SW06, SW6
333 .EQUIV SW05, SW5
334 .EQUIV SW04, SW4
335 .EQUIV SW03, SW3
336 .EQUIV SW02, SW2
337 .EQUIV SW01, SW1
338 .EQUIV SW00, SW0

```

```

340 ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
341 100000 BIT15= 100000
342 040000 BIT14= 40000
343 020000 BIT13= 20000
344 010000 BIT12= 10000
345 004000 BIT11= 4000
346 002000 BIT10= 2000
347 001000 BIT09= 1000
348 000400 BIT08= 400
349 000200 BIT07= 200
350 000100 BIT06= 100
351 000040 BIT05= 40
352 000020 BIT04= 20
353 000010 BIT03= 10
354 000004 BIT02= 4
355 000002 BIT01= 2
356 000001 BIT00= 1
357 .EQUIV BIT09, BIT9
358 .EQUIV BIT08, BIT8
359 .EQUIV BIT07, BIT7
360 .EQUIV BIT06, BIT6
361 .EQUIV BIT05, BIT5
362 .EQUIV BIT04, BIT4
363 .EQUIV BIT03, BIT3
364 .EQUIV BIT02, BIT2
365 .EQUIV BIT01, BIT1
366 .EQUIV BIT00, BIT0

```

```

367 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
368 ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
369 RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
370 TBITVEC= 14 ;: "T" BIT
371 TRTVEC= 14 ;: TRACE TRAP
372 BPTVEC= 14 ;: BREAKPOINT TRAP (BPT)
373 IOTVEC= 20 ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
374 PWRVEC= 24 ;: POWER FAIL
375 EMTVEC= 30 ;: EMULATOR TRAP (EMT) **ERROR**
376 TRAPVEC= 34 ;: "TRAP" TRAP
377 TKVEC= 60 ;: TTY KEYBOARD VECTOR
378 TPVEC= 64 ;: TTY PRINTER VECTOR
379 CACHVEC= 114 ;: CACHE ERROR INTERRUPT VECTOR
380

```

```

381      00024r      PIRQVEC=240      ;;PROGRAM INTERRUPT REQUEST VECTOR
382      00025U      MMVEC= 250      ;;MEMORY MANAGEMENT VECTOR
383
384      .SBTTL  CACHE  REGISTER DEFINITIONS
385
386      177740      LOADRS = 177740      ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
387      177742      HIADRS = 177742      ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
388      177744      MEMERR = 177744      ;;CACHE ERROR REGISTER
389      177746      CONTRL = 177746      ;;MEMORY CONTROL REGISTER
390      177750      MAINT = 177750      ;;MEMORY MAINTENANCE REGISTER
391      177752      HITMIS = 177752      ;;HIT MISS REGISTER "1" IMPLIES HIT IN CACHE
392
393      .SBTTL  CPU REGISTER DEFINITIONS
394
395
396
397
398      177760      SIZELO = 177760      ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
399      177762      SIZEHI = 177762      ;;TO GET TO THE LAST 32 WORDS OF MEMORY
400      177764      SYSTID = 177764      ;;HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
401      177766      CPUERR = 177766      ;;CURRENTLY ALL ZERO
402      177766      CPUERR = 177766      ;;SYSTEM ID REGISTER
403      177766      CPUERR = 177766      ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
404      177766      CPUERR = 177766      ;;THE TRAP TO ERRVEC (000004)
405
406
407
408
409      .SBTTL  MEMORY MANAGEMENT DEFINITIONS
410
411
412      ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
413
414      177572      MMR0= 177572
415      177574      MMR1= 177574
416      177576      MMR2= 177576
417      172516      MMR3= 172516
418      .EQUIV MMR0,SR0
419      .EQUIV MMR1,SR1
420      .EQUIV MMR2,SR2
421      .EQUIV MMR3,SR3
422
423      ;*USER "I" PAGE DESCRIPTOR REGISTERS
424
425      177600      UIPDR0= 177600
426      177602      UIPDR1= 177602
427      177604      UIPDR2= 177604
428      177606      UIPDR3= 177606
429      177610      UIPDR4= 177610
430      177612      UIPDR5= 177612
431      177614      UIPDR6= 177614
432      177616      UIPDR7= 177616
433
434      ;*USER "D" PAGE DESCRIPTOR REGISTERS
435
436      177620      UDPDR0= 177620
    
```

437	177622	UDPDR1= 177622
438	177624	UDPDR2= 177624
439	177626	UDPDR3= 177626
440	177630	UDPDR4= 177630
441	177632	UDPDR5= 177632
442	177634	UDPDR6= 177634
443	177636	UDPDR7= 177636
444		
445		;*USER "I" PAGE ADDRESS REGISTERS
446		
447	177640	UIPAR0= 177640
448	177642	UIPAR1= 177642
449	177644	UIPAR2= 177644
450	177646	UIPAR3= 177646
451	177650	UIPAR4= 177650
452	177652	UIPAR5= 177652
453	177654	UIPAR6= 177654
454	177656	UIPAR7= 177656
455		
456		;*USER "D" PAGE ADDRESS REGISTERS
457		
458	177660	UDPAR0= 177660
459	177662	UDPAR1= 177662
460	177664	UDPAR2= 177664
461	177666	UDPAR3= 177666
462	177670	UDPAR4= 177670
463	177672	UDPAR5= 177672
464	177674	UDPAR6= 177674
465	177676	UDPAR7= 177676
466		
467		;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
468		
469	172200	SIPDR0= 172200
470	172202	SIPDR1= 172202
471	172204	SIPDR2= 172204
472	172206	SIPDR3= 172206
473	172210	SIPDR4= 172210
474	172212	SIPDR5= 172212
475	172214	SIPDR6= 172214
476	172216	SIPDR7= 172216
477		
478		;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
479		
480	172220	SDPDR0= 172220
481	172222	SDPDR1= 172222
482	172224	SDPDR2= 172224
483	172226	SDPDR3= 172226
484	172230	SDPDR4= 172230
485	172232	SDPDR5= 172232
486	172234	SDPDR6= 172234
487	172236	SDPDR7= 172236
488		
489		;*SUPERVISOR "I" PAGE ADDRESS REGISTERS
490		
491	172240	SIPARO= 172240
492	172242	SIPAR1= 172242

493	172244	SIPAR2= 172244
494	172246	SIPAR3= 172246
495	172250	SIPAR4= 172250
496	172252	SIPAR5= 172252
497	172254	SIPAR6= 172254
498	172256	SIPAR7= 172256

;*SUPERVISOR "D" PAGE ADDRESS REGISTERS

500		
501		
502	172260	SDPAR0= 172260
503	172262	SDPAR1= 172262
504	172264	SDPAR2= 172264
505	172266	SDPAR3= 172266
506	172270	SDPAR4= 172270
507	172272	SDPAR5= 172272
508	172274	SDPAR6= 172274
509	172276	SDPAR7= 172276

;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

510		
511		
512		
513	172300	KIPDR0= 172300
514	172302	KIPDR1= 172302
515	172304	KIPDR2= 172304
516	172306	KIPDR3= 172306
517	172310	KIPDR4= 172310
518	172312	KIPDR5= 172312
519	172314	KIPDR6= 172314
520	172316	KIPDR7= 172316

;*KERNEL "D" PAGE DESCRIPTOR REGISTERS

521		
522		
523		
524	172320	KDPDR0= 172320
525	172322	KDPDR1= 172322
526	172324	KDPDR2= 172324
527	172326	KDPDR3= 172326
528	172330	KDPDR4= 172330
529	172332	KDPDR5= 172332
530	172334	KDPDR6= 172334
531	172336	KDPDR7= 172336

;*KERNEL "I" PAGE ADDRESS REGISTERS

532		
533		
534		
535	172340	KIPAR0= 172340
536	172342	KIPAR1= 172342
537	172344	KIPAR2= 172344
538	172346	KIPAR3= 172346
539	172350	KIPAR4= 172350
540	172352	KIPAR5= 172352
541	172354	KIPAR6= 172354
542	172356	KIPAR7= 172356

;*KERNEL "D" PAGE ADDRESS REGISTERS

543		
544		
545		
546	172360	KDPAR0= 172360
547	172362	KDPAR1= 172362
548	172364	KDPAR2= 172364

549	172366	KDPAR3= 172366
550	172370	KDPAR4= 172370
551	172372	KDPAR5= 172372
552	172374	KDPAR6= 172374
553	172376	KDPAR7= 172376

.SBTTL UNIBUS MAP REGISTER DEFINITIONS

;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'

565	170200	MAPL00 = 170200
566	170202	MAPH00 = 170202
567	170204	MAPL01 = 170204
568	170206	MAPH01 = 170206
569	170210	MAPL02 = 170210
570	170212	MAPH02 = 170212
571	170214	MAPL03 = 170214
572	170216	MAPH03 = 170216
573	170220	MAPL04 = 170220
574	170222	MAPH04 = 170222
575	170224	MAPL05 = 170224
576	170226	MAPH05 = 170226
577	170230	MAPL06 = 170230
578	170232	MAPH06 = 170232
579	170234	MAPL07 = 170234
580	170236	MAPH07 = 170236
581	170240	MAPL10 = 170240
582	170242	MAPH10 = 170242
583	170244	MAPL11 = 170244
584	170246	MAPH11 = 170246
585	170250	MAPL12 = 170250
586	170252	MAPH12 = 170252
587	170254	MAPL13 = 170254
588	170256	MAPH13 = 170256
589	170260	MAPL14 = 170260
590	170262	MAPH14 = 170262
591	170264	MAPL15 = 170264
592	170266	MAPH15 = 170266
593	170270	MAPL16 = 170270
594	170272	MAPH16 = 170272
595	170274	MAPL17 = 170274
596	170276	MAPH17 = 170276
597	170300	MAPL20 = 170300
598	170302	MAPH20 = 170302
599	170304	MAPL21 = 170304
600	170306	MAPH21 = 170306
601	170310	MAPL22 = 170310
602	170312	MAPH22 = 170312
603	170314	MAPL23 = 170314
604	170316	MAPH23 = 170316

605	170320	MAPL24 =	170320
606	170320	MAPH24 =	170320
607	170324	MAPL25 =	170324
608	170326	MAPH25 =	170326
609	170330	MAPL26 =	170330
610	170332	MAPH26 =	170332
611	170334	MAPL27 =	170334
612	170336	MAPH27 =	170336
613	170340	MAPL30 =	170340
614	170342	MAPH30 =	170342
615	170344	MAPL31 =	170344
616	170346	MAPH31 =	170346
617	170350	MAPL32 =	170350
618	170352	MAPH32 =	170352
619	170354	MAPL33 =	170354
620	170356	MAPH33 =	170356
621	170360	MAPL34 =	170360
622	170362	MAPH34 =	170362
623	170364	MAPL35 =	170364
624	170366	MAPH35 =	170366
625	170370	MAPL36 =	170370
626	170372	MAPH36 =	170372
627	170374	MAPL37 =	170374
628	170376	MAPH37 =	170376
629		.EQUIV	MAPL00, MAPL0
630		.EQUIV	MAPH00, MAPH0
631		.EQUIV	MAPL01, MAPL1
632		.EQUIV	MAPH01, MAPH1
633		.EQUIV	MAPL02, MAPL2
634		.EQUIV	MAPH02, MAPH2
635		.EQUIV	MAPL03, MAPL3
636		.EQUIV	MAPH03, MAPH3
637		.EQUIV	MAPL04, MAPL4
638		.EQUIV	MAPH04, MAPH4
639		.EQUIV	MAPL05, MAPL5
640		.EQUIV	MAPH05, MAPH5
641		.EQUIV	MAPL06, MAPL6
642		.EQUIV	MAPH06, MAPH6
643		.EQUIV	MAPL07, MAPL7
644		.EQUIV	MAPH07, MAPH7
645			
646			
647			
648			
649			

650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687

000000
000200
000000 000137 001102
000046
000052
000204

.SBTTL TRAP CATCHER
.=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
.SBTTL STARTING ADDRESS(ES)
.=200
JMP @BEGIN ;; JUMP TO STARTING ADDRESS OF PROGRAM
:*****
.SBTTL ACT11 HOOKS
:*THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11
:*
:*LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOGICAL
:*END OF THE PROGRAM.
:*LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
:*AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS
:*TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:
:*
:* BIT 15=1 PROGRAM SHOULD BE POWER FAILED WHILE RUNNING
:* =0 NO POWER FAIL DESIRED
:*
:* BIT 14=1 PROGRAM RUN TIME IS MEMORY SIZE DEPENDENT
:* =0 RUN TIME IS NOT MEMORY SIZE DEPENDENT
:*
:* BITS 13-0 MUST BE ZERO'S
\$SVPC=. ;; SAVE LOCATION COUNTER
.=46 ;; SET LOCATION COUNTER
.WORD SENDAD ;; SET LOC.46 TO ADDRESS SENDAD
.=52 ;; SET LOCATION COUNTER
.WORD BIT14 ;; SET LOC.52 TO BIT14
.= \$SVPC ;; RESTORE LOCATION COUNTER

```

688          001100          . = 1100
689
690 001100 006405          FACTOR: .WORD 20000./6          ;20000 E-7/6 E-7 = 2MS/.6NS =
691                                     ;POWER DOWN TIME DIVIDED BY "SOB" EXECUTION TIME
692          001102          BEGIN:
693 001102 012706 001100          MOV      #STACK,SP          ;;SETUP THE STACK POINTER
694 001106 012737 004440 000034          MOV      #STRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
695 001114 012737 000340 000036          MOV      #340,#TRAPVEC+2;LEVEL 7
696 001122 005067 002020          CLR      SPASS          ;;CLEAR THE PASS COUNT
697 001126 016767 001764 001754          MOV      SENDCT,SEOPCT ;;SETUP END-OF-PROGRAM COUNTER
698 001134 012777 003472 002506          MOV      #POWDWN,#DVEC  ;SET UP POWER DOWN VECTOR
699 001142 012737 003372 000114          MOV      #PARERR,#CACHVEC;SET UP PARITY ERROR VECTOR
700 001150 012737 000001 003374          MOV      #1,#PARFLG    ;INITIALIZE THE MULTI PARITY ERROR INDICATOR
701
702                                     ;DETERMINE THE NUMBER OF 4K BLOCKS MEMORY IS MADE UP OF.
703
704 001156 005000          CLR      R0          ;MAKE MSB'S = 0
705 001160 013701 177760          MOV      #SIZELO,R1   ;GET PAR VALUE FOR LAST 32 WORDS OF
706 001164 005201          INC      R1          ;MEMORY AND ROUND UP TO A 4K BOUNDARY
707 001166 071027 000200          DIV      #200,R0     ;FORM THE BLOCK NUMBER OF
708 001172 005200          DEC      R0          ;THE LAST 4K BLOCK AND
709 001174 010067 002460          MOV      R0,LIMIT    ;SAVE IT.
710
711                                     ;IDENTIFY THE PROGRAM AND TYPE INSTRUCTIONS
712
713 001200 005227 177777          INC      #-1          ;;FIRST TIME?
714 001204 001116          BNE     TST1          ;;BRANCH IF NO
715 001206 022737 003132 000042          CMP      #SENDAD,#42 ;;ACT-11?
716 001214 001512          BEQ     TST1          ;;BRANCH IF YES
717 001216 104400 001224          TYPE    ,65$         ;;TYPE ASCIZ STRING
718 001222 000413          BR      64$          ;;GET OVER THE ASCIZ
719                                     ;;65$: .ASCIZ <CRLF>#MAINDEC-11-DEK8G-A*<CRLF>
720 001252                                     64$:
721 001252 104400 001260          TYPE    ,67$         ;;TYPE ASCIZ STRING
722 001256 000406          BR      66$          ;;GET OVER THE ASCIZ
723                                     ;;67$: .ASCIZ <CRLF>#BANKS 0 - *
724                                     66$:
725 001274 010046          MOV      R0,-(SP)    ;;SAVE R0 FOR TYPEOUT
726 001276 104404          TYPOS   ;GO TYPE--OCTAL ASCII
727 001300          .BYTE  6           ;;TYPE 6 DIGITS
728 001301          .BYTE  0           ;;SUPPRESS LEADING ZEROS
729 001302 104400 001310          TYPE    ,69$         ;;TYPE ASCIZ STRING
730 001306 000404          BR      68$          ;;GET OVER THE ASCIZ
731                                     ;;69$: .ASCIZ * EXIST*
732                                     68$:
733 001320 104400 001326          TYPE    ,71$         ;;TYPE ASCIZ STRING
734 001324 000423          BR      70$          ;;GET OVER THE ASCIZ
735                                     ;;71$: .ASCIZ <CRLF><LF>#INTERRUPT THE POWER AFTER THE TEST*
736 001374                                     70$:
737 001374 104400 001402          TYPE    ,73$         ;;TYPE ASCIZ STRING
738 001400 000420          BR      72$          ;;GET OVER THE ASCIZ
739                                     ;;73$: .ASCIZ * NUMBER APPEARS IN THE DISPLAY*<CRLF>
740 001442                                     72$:

```



```

741
742
743
744 001442
745 001442 012737 000001 177570
746 001450 005037 177776
747 001454 012703 001462
748 001460 000001
749 001462 010600 2S:
750 001464 022700 001074
751 001470 001401
752 001472 000000
753 001474 012706 001100
754 001500 013700 001074
755 001504 022700 001462
756 001510 001401
757 001512 000000
758 001514 013700 001076
759 001520 022700 000000
760 001524 001401
761 001526 000000
762 001530
763 001530 032737 040000 177570 1S:
764 001536 001341
765
766
767
768
769
770 001540
771 001540 012737 000002 177570
772 001546 012737 040000 177776
773 001554 012703 001562
774 001560 000001
775 001562 012706 001100 2S:
776 001566 013700 001074
777 001572 022700 001562
778 001576 001401
779 001600 000000
780 001602 013700 001076
781 001606 022700 040000
782 001612 001401
783 001614 000000
784 001616
785 001616 032737 040000 177570 1S:
786 001624 001345

```

```

*****
*TEST 1      SIMPLE DOWN/UP TEST (KERNAL)
*****
TST1:
MOV      #1, @DISPLAY      ;SET TEST NUMBER
CLR      @PS                ;SET KERNAL MODE
MOV      #2S, R3           ;SET POWER UP RETURN
WAIT                    ;WAIT FOR THE POWER FAIL
2S:      MOV      SP, R0     ;GET SP
          CMP      @STACK-4, R0 ;CHECK SP
          BEQ      .+4       ;SKIP IF OK
          HALT                    ;SP NOT "STACK-4"
          MOV      @STACK, SP  ;RESET SP
          MOV      @STACK-4, R0 ;GET RETURN ADDRESS
          CMP      #2S, R0     ;CHECK ADDRESS
          BEQ      .+4       ;SKIP IF OK
          HALT                    ;ADDRESS ON STACK IS WRONG
          MOV      @STACK-2, R0 ;GET OLD PS
          CMP      #0, R0      ;CHECK OLD PS
          BEQ      .+4       ;SKIP IF OK
          HALT                    ;OLD PS IS WRONG
1S:      BIT      #SW14, @SWR  ;LOOP ON TEST?
          BNE      TST1       ;LOOP TO TST1
*****
*TEST 2      SIMPLE DOWN/UP TEST (SUPERVISOR)
*****
TST2:
MOV      #2, @DISPLAY      ;SET TEST NUMBER
MOV      #40000, @PS       ;SET SUPERVISOR MODE
MOV      #2S, R3           ;SET POWER UP RETU.
WAIT                    ;WAIT FOR THE POWER FAIL
2S:      MOV      @STACK, SP  ;RESET SP
          MOV      @STACK-4, R0 ;GET RETURN ADDRESS
          CMP      #2S, R0     ;CHECK ADDRESS
          BEQ      .+4       ;SKIP IF OK
          HALT                    ;ADDRESS ON STACK IS WRONG
          MOV      @STACK-2, R0 ;GET OLD PS
          CMP      #40000, R0  ;CHECK OLD PS
          BEQ      .+4       ;SKIP IF OK
          HALT                    ;OLD PS IS WRONG
1S:      BIT      #SW14, @SWR  ;LOOP ON TEST?
          BNE      TST2       ;LOOP TO TST2

```

E02

MAINDEC-11-DEABG-A
DEABGA.P11 T3

PDP-11/70 POWER FAIL MACY11 27(732) 20-SEP-76 13:27 PAGE 16
SIMPLE DOWN/UP TEST (USER)

```

787
788
789
790 001626
791 001626 012737 000003 177570
792 001634 012737 140000 177776
793 001642 012703 001650
794 001646 000001
795 001650 012706 001100 2S:
796 001654 013700 001074
797 001660 022700 001650
798 001664 001401
799 001666 000000
800 001670 013700 001076
801 001674 022700 140000
802 001700 001401
803 001702 000000
804 001704
805 001704 032737 040000 177570 1S:
806 001712 001345
807
808
809
810
811
812
813 001714
814 001714 012737 000004 177570
815 001722 005037 177776
816 001726 012737 001740 000004
817 001734 012703 001752
818 001740 012706 001100 3S:
819 001744 005737 000003
820 001750 000000
821 001752 012737 000006 000004 1S:
822 001760 032737 040000 177570
823 001766 001352

```

```

*****
;TEST 3      SIMPLE DOWN/UP TEST (USER)
*****
TST3:
MOV      #3,0#DISPLAY      ;SET TEST NUMBER
MOV      #140000,0#PS      ;SET USER MODE
MOV      #2$,R3            ;SET POWER UP RETURN
WAIT                     ;WAIT FOR THE POWER FAIL
2S:      MOV      #STACK,SP ;RESET SP
MOV      0#STACK-4,R0     ;GET RETURN ADDRESS
CMP      #2$,R0           ;CHECK ADDRESS
BEQ      .+4              ;SKIP IF OK
HALT                     ;ADDRESS ON STACK IS WRONG
MOV      0#STACK-2,R0     ;GET OLD PS
CMP      #140000,R0       ;CHECK OLD PS
BEQ      .+4              ;SKIP IF OK
HALT                     ;OLD PS IS WRONG

1S:      BIT      #SW14,0#SWR ;LOOP ON TEST?
BNE     TST3              ;LOOP TO TST3

*****
;TEST 4      POWER FAIL WITH ODD ADDRESS
*****
TST4:
MOV      #4,0#DISPLAY      ;SET TEST NUMBER
CLR      0#PS              ;SET KERNAL MODE
MOV      #3$,0#ERRVEC     ;SET TRAP VECTOR
MOV      #1$,R3           ;SET RETURN ADDRESS FOR POWER FAIL
3S:      MOV      #STACK,SP ;RESET STACK
TST      0#3              ;CAUSE ODD ADDRESS TRAP
HALT                     ;ODD ADDRESS TRAP FAILED
1S:      MOV      #6,0#ERRVEC ;RESET 4
BIT      #SW14,0#SWR     ;LOOP ON TEST?
BNE     TST4              ;LOOP TO TST4

```

F02

MAINDEC-11-DEK8G-A
DEK8GA.P11 TS

POP-11/70 POWER FAIL MACY11 27(732) 20-SEP-76 13:27 PAGE 17
POWER FAIL IN THE RED ZONE

```

824
825
826
827 001770
828 001770 012737 000005 177570
829 001776 005037 177776
830 002002 012737 002022 000004
831 002010 012703 002040
832 002014 012706 000002
833 002020 000001
834 002022 012777 002030 001614 25:
835 002030 000000 75:
836 002032 012777 003472 001610
837 002040 012706 001100 15:
838 002044 012737 000006 000004
839 002052 013700 000002
840 002056 005737 000002
841 002062 001401
842 002064 000000
843 002066 013700 000000
844 002072 022737 003472 000000
845 002100 001401
846 002102 000000
847 002104 032737 040000 177570
848 002112 001326
849
850
851
852
853
854 002114
855 002114 012737 000006 177570
856 002122 012737 002134 000004
857 002130 012703 002152
858 002134 012706 001100 35:
859 002140 005037 177776
860 002144 010037 173000
861 002150 000000
862 002152 012706 001100 15:
863 002156 012737 000006 000004
864 002164 032737 040000 177570
865 002172 001350

```

```

*****
*TEST 5 POWER FAIL IN THE RED ZONE
*****
TST5:
MOV #5,0#DISPLAY ;SET TEST NUMBER
CLR 0#PS ;SET KERNAL MODE
MOV #25,0#ERRVEC ;SET TRAP REGISTER
MOV #15,R3 ;SET POWER UP RETURN
MOV #2,5P ;SET STACK TO RED ZONE
WAIT ;WAIT FOR POWER FAIL TRAP
MOV #75,0UVEC ;SET UVEC TO HALT
HALT ;ILLEGAL TRAP TO 4
MOV #POWDN,0DVEC ;RESET DVEC
MOV #STACK,5P ;RESET STACK
MOV #6,0#ERRVEC ;RESET 4
MOV 0#2,R0 ;GET FOR TYPING
TST 0#2 ;IS 2 OK?
BEQ .+4 ;SKIP IF OK
HALT ;NO!
MOV 0#0,R0 ;GET FOR TYPING
CMP #POWDN,0#0 ;IS 0 OK?
BEQ .+4 ;SKIP IF OK
HALT ;0 IS WRONG!
BIT #SW14,0#SWR ;LOOP ON TEST?
BNE TST5 ;LOOP TO TST5

```

```

*****
*TEST 6 POWER FAIL WITH TIME OUT (KERNAL)
*****
TST6:
MOV #6,0#DISPLAY ;SET TEST NUMBER
MOV #35,0#ERRVEC ;SET TRAP VECTOR
MOV #15,R3 ;SET UP RETURN ADDRESS FOR POWER FAIL
MOV #STACK,5P ;SET STACK
CLR 0#PS ;SET KERNAL MODE
MOV R0,0#173000 ;CAUSE A TIMEOUT
HALT ;TIMEOUT FAILED
MOV #STACK,5P ;SET STACK
MOV #6,0#ERRVEC ;RESET 4
BIT #SW14,0#SWR ;LOOP ON TEST?
BNE TST6 ;LOOP TO TST6

```

```

866
867
868
869 002174
870 002174 012737 000007 177570
871 002202 005037 177776
872 002206 005067 001444
873 002212 012737 002236 000004
874 002220 012706 000400
875 002224 012703 002232
876 002230 000001
877 002232 000000 1S:
878 002234 000422
879 002236 012737 000006 000004 2S:
880 002244 005767 001406
881 002250 001010
882 002252 012777 002250 001364
883 002260 000000 7S:
884 002262 012777 003472 001360
885 002270 000404
886 002272 012703 002302 5S:
887 002276 000002
888 002300 000000
889 002302 4S:
890 002302 032737 040000 177570
891 002310 001331
892
893
894
895
896
897 002312
898 002312 012737 000010 177570
899 002320 005037 177776
900 002324 012703 002344
901 002330 012706 001100
902 002334 000005 3S:
903 002336 000005
904 002340 000005
905 002342 000774
906 002344 012706 001100 1S:
907 002350 032737 040000 177570
908 002356 001355

```

```

*****
;TEST 7 POWER FAIL IN THE YELLOW ZONE (KERNAL)
*****
TST7:
MOV #7, D#DISPLAY ;SET TEST NUMBER
CLR D#PS ;SET KERNAL MODE
CLR FLAG ;CLEAR THE FLAG
MOV #2S, D#ERRVEC ;SET SICK TPAP ADDRESS
MOV #400, SP ;SET STACK TO YELLOW ZONE
MOV #1S, R3 ;SET RETURN ADDRESS FOR POWER FAIL
WAIT ;WAIT FOR POWER FAIL
HALT ;POWER FAIL RETURNED TOO SOON
BR 4S ;SKIP SP CHECK
MOV #6, D#ERRVEC ;RESET 4
TST FLAG ;IS THE FIRST INSTRUCTION FLAG SET?
BNE 5S ;YES
MOV #7S, D#UVEC ;SET UVEC TO HALT
HALT ;NOT ENOUGH OR TOO MANY INSTR. EXEC.
MOV #POWDWN, D#DVEC ;SET DVEC
BR 4S ;GET OUT
MOV #4S, R3 ;SET RETURN
RTI ;GO TO THE POWER FAIL ROUTINE
HALT ;SHOULD NOT RETURN HERE
BIT #SW14, D#SWR ;LOOP ON TEST?
BNE TST7 ;LOOP TO TST7

```

```

*****
;TEST 10 POWER FAIL WITH RESETS
*****
TST10:
MOV #10, D#DISPLAY ;SET TEST NUMBER
CLR D#PS ;SET KERNAL MODE
MOV #1S, R3 ;SET RETURN ADDRESS
MOV #STACK, SP ;RESET STACK
RESET ;RESETS
RESET ;TO WAIT
RESET ;IN
BR 3S ;LOOP
MOV #STACK, SP ;RESET STACK
BIT #SW14, D#SWR ;LOOP ON TEST?
BNE TST10 ;LOOP TO TST10

```

H02

MAINDEC-11-DEKBG-A
DEKBGA.P11 T11

PDP-11/70 POWER FAIL MACY11 27(732) 20-SEP-76 13:27 PAGE 19
POWER FAIL WITH ODD ADDRESS (SUPERVISOR)

```

909
910
911
912 002360
913 002360 012737 000011 177570
914 002366 012737 002400 000004
915 002374 012703 002424
916 002400 012706 001100
917 002404 012737 040000 177776
918 002412 005737 000003
919 002416 005037 1777.6
920 002422 000000
921 002424 012706 001100
922 002430 012737 000006 000004
923 002436 032737 040000 177570
924 002444 001345
925
926
927
928
929
930 002446
931 002446 012737 000012 177570
932 002454 012737 002466 000004
933 002462 012703 002512
934 002466 012706 001100
935 002472 012737 040000 177776
936 002500 010037 173000
937 002504 005037 177776
938 002510 000000
939 002512 012706 001100
940 002516 012737 000006 000004
941 002524 032737 040000 177570
942 002532 001345

```

```

*****
;*TEST 11 POWER FAIL WITH ODD ADDRESS (SUPERVISOR)
*****
TST11:
MOV #11,0#DISPLAY ;SET TEST NUMBER
MOV #3$,0#ERRVEC ;SET TRAP VECTOR
MOV #1$,R3 ;SET RETURN ADDRESS FOR POWER FAIL
3$: MOV #STACK,SP ;RESET STACK
MOV #40000,0#PS ;SET SUPERVISOR MODE
TST 0#3 ;CAUSE ODD ADDRESS TRAP
CLR 0#PS ;SET KERNAL MODE
HALT ;ODD ADDRESS TRAP FAILED
1$: MOV #STACK,SP ;RESET STACK POINTER
MOV #6,0#ERRVEC ;RESET 4
BIT #SW14,0#SWR ;LOOP ON TEST?
BNE TST11 ;LOOP TO TST11

```

```

*****
;*TEST 12 POWER FAIL WITH TIME OUT (SUPERVISOR)
*****
TST12:
MOV #12,0#DISPLAY ;SET TEST NUMBER
MOV #3$,0#ERRVEC ;SET TRAP VECTOR
MOV #1$,R3 ;SET UP RETURN ADDRESS FOR POWER FAIL
3$: MOV #STACK,SP ;RESET STACK
MOV #40000,0#PS ;SET SUPERVISOR MODE
MOV R0,0#173000 ;CAUSE A TIMEOUT
CLR 0#PS ;SET KERNAL MODE
HALT ;TIMEOUT FAILED
1$: MOV #STACK,SP ;RESET STACK
MOV #6,0#ERRVEC ;RESET 4
BIT #SW14,0#SWR ;LOOP ON TEST?
BNE TST12 ;LOOP TO TST12

```

```

943
944
945
946 002534
947 002534 012737 000013 177570
948 002542 012737 002554 000004
949 002550 012703 002600
950 002554 012706 001100
951 002560 012737 140000 177776
952 002566 005737 000003
953 002572 005037 177776
954 002576 000000
955 002600 012706 001100
956 002604 012737 000006 000004
957 002612 032737 040000 177570
958 002620 001345
959
960
961
962
963
964
965 002622
966 002630 0127 000014 177570
967 002636 0127 002642 000004
968 002642 012703 002666
969 002646 012706 001100
970 002654 012737 140000 177776
971 002660 010037 173000
972 002664 005037 177776
973 002666 000000
974 002666 012706 001100
975 002672 012737 000006 000004
976 002700 032737 040000 177570
977 002706 001345

```

```

*****
;*TEST 13 POWER FAIL WITH ODD ADDRESS (USER)
*****
TST13:

```

```

MOV #13,0#DISPLAY ;SET TEST NUMBER
MOV #3$,0#ERRVEC ;SET TRAP VECTOR
MOV #1$,R3 ;SET RETURN ADDRESS FOR POWER FAIL
3$: MOV #STACK,SP ;RESET STACK
MOV #140000,0#PS ;SET USER MODE
TST 0#3 ;CAUSE ODD ADDRESS TRAP
CLR 0#PS ;SET KERNAL MODE
HALT ;ODD ADDRESS TRAP FAILED
1$: MOV #STACK,SP ;RESET SP
MOV #6,0#ERRVEC ;RESET 4
BIT #SW14,0#SWR ;LOOP ON TEST?
BNE TST13 ;LOOP TO TST13

```

```

*****
;*TEST 14 POWER FAIL WITH TIME OUT (USER)
*****
TST14:

```

```

MOV #14,0#DISPLAY ;SET TEST NUMBER
MOV #3$,0#ERRVEC ;SET TRAP VECTOR
MOV #1$,R3 ;SET UP RETURN ADDRESS FOR POWER FAIL
3$: MOV #STACK,SP ;RESET STACK
MOV #140000,0#PS ;SET USER MODE
MOV R0,0#173000 ;CAUSE A TIMEOUT
CLR 0#PS ;SET KERNAL MODE
HALT ;TIMEOUT FAILED
1$: MOV #STACK,SP ;RESET STACK
MOV #6,0#ERRVEC ;RESET 4
BIT #SW14,0#SWR ;LOOP ON TEST?
BNE TST14 ;LOOP TO TST14

```

```

977
978
979
980 002710
981 002710 012737 000015 177570
982 002716 005037 177776
983 002722 012737 002762 000004
984 002730 004767 000730
985 002734 012737 002752 000250
986 002742 012703 002764
987 002746 005237 177572
988 002752 012706 001100 3$:
989 002756 005237 140000
990 002762 000000 4$:
991
992 002764 005037 177572 1$:
993 002770 012706 001100 2$:
994 002774 012737 000006 000004
995 003002 032737 040000 177570
996 003010 001337
997
998
999
1000
1001 003012 000240
1002 003014 005037 177776
1003 003020 012702 000010
1004 003024 004767 000120 4$:
1005 003030 012703 003050
1006 003034 012737 000016 177570
1007 003042 004767 000206 2$:
1008 003046 000775
1009 003050 012706 001100 1$:
1010 003054 004767 000174
1011 003060 077217
1012 003062 032737 040000 177570
1013 003070 001350

```

```

*****
;TEST 15 MEMORY MANAGEMENT ABORT TEST
*****
TST15:
MOV #15, D#DISPLAY ;SET TEST NUMBER
CLR D#PS ;SET KERNAL MODE
MOV #4$, D#ERRVEC ;SET FOR TIMEOUT
JSR PC, MAP ;MAP THE WORLD
MOV #3$, D#MMVEC ;SET MEMORY MANAGEMENT VECTOR
MOV #1$, R3 ;LOAD PF RETURN
INC D#MMRO ;TURN MEMORY MANAGEMENT ON
MOV #STACK, SP ;ZAP STACK
INC D#140000 ;ACCESS VIOLATION
HALT ;NO VIOLATION OR TRAP TO 4

1$: CLR D#MMRO ;TURN OFF MEMORY MANAGEMENT
2$: MOV #STACK, SP ;MAKE A NEW STACK
MOV #6, D#ERRVEC ;RESET 4
BIT #SW14, D#SWR ;LOOP ON TEST?
BNE TST15 ;LOOP TO TST15

*****
;TEST 16 MEMORY VOLATILITY TEST
*****
TST16: NOP
CLR D#PS ;SET KERNAL MODE
MOV #10, R2 ;LOAD COUNT OF TEST ITERATIONS
4$: JSR PC, LOAD ;LOAD ALL MEMORY WITH 52525
MOV #1$, R3 ;POWER FAIL RETURN ADDRESS
MOV #16, D#DISPLAY ;SET TEST NUMBER
2$: JSR PC, CHECK ;CHECK FOR THE 52525
BR 2$ ;LOOP FOR EVER OR POWER FAIL
1$: MOV #STACK, SP ;ZAP THE STACK
JSR PC, CHECK ;CHECK ALL MEMORY
SOB R2, 4$ ;DO IT 10 TIMES
BIT #SW14, D#SWR ;LOOP ON TEST?
BNE TST16 ;LOOP TO TST16

```

K02

MAINDEC-11-DEKBG-A
DEKPGA.P11 T16

PDP-11/70 POWER FAIL
MEMORY VOLATILITY TEST

MACY11 27(732) 20-SEP-76 13:27 PAGE 22

```

1014 ;:*****
1015
1016 .SBTTL END OF PASS ROUTINE
1017
1018 ;*INCREMENT THE PASS NUMBER ($PASS)
1019 ;*IF THERES A MONITOR GO TO IT
1020 ;*IF THERE ISN'T JUMP TO 200
1021
1022 $EOP:
1023 003072 000240 NOP
1024 003074 005267 000046 INC $PASS ;: INCREMENT THE PASS NUMBER
1025 003100 042767 100000 000040 BIC #10000,$PASS ;: DON'T ALLOW A NEG. NUMBER
1026 003106 005327 DEC (PC)+ ;: LOOP?
1027 003110 000001 $EOPCT: .WORD 1
1028 003112 003013 BGT $DOAGN ;: YES
1029 003114 012737 MOV (PC)+,2(PC)+ ;: RESTORE COUNTER
1030 003116 000001 $ENDCT: .WORD 1
1031 003120 003110 $EOPCT
1032 003122 013700 000042 $GET42: MOV 2*42,R0 ;: GET MONITOR ADDRESS
1033 003126 001405 BEQ $DOAGN ;: BRANCH IF NO MONITOR
1034 003130 000005 RESET ;: CLEAR THE WORLD
1035 003132 004710 $ENDAD: JSR PC,(R0) ;: GO TO MONITOR
1036 003134 000240 NOP ;: SAVE ROOM
1037 003136 000240 NOP ;: FOR
1038 003140 000240 NOP ;: ACT11
1039 003142 $DOAGN:
1040 003142 000137 000200 JMP 2*200 ;: RETURN
1041 003146 000000 $PASS: .WORD 0 ;: NUMBER OF PASSES

```


1042									
1043						.SBTTL	LOAD MEMORY WITH DATA PATTERN		
1044	003150	004767	000510			LOAD:	JSR PC,MAP	;SET UP MEMORY MANAGEMENT REGISTERS	
1045	003154	016704	000500				MOV LIMIT,R4	;GET BANK COUNT	
1046	003160	016705	000476				MOV DATA,R5	;GET THE DATA PATTERN	
1047	003164	012737	003214	000250			MOV #25,MMVEC	;SET UP FOR MEMORY MANAGEMENT ABORTS	
1048	003172	012700	004470				MOV #END,R0	;FORM ADDRESS OF WHERE TO START WRITING THE DATA	
1049	003176	062700	120000				ADD #120000,R0	;USE KIPARS FOR ADDRESSING	
1050	003202	012737	000001	177572			MOV #1,MMRO	;TURN ON MEMORY MANAGEMENT	
1051	003210	010520			1\$:		MOV R5,(R0)+	;WRITE THE DATA	
1052	003212	000776					BR 1\$		
1053	003214	005304			2\$:		DEC R4	;LAST BANK WRITTEN?	
1054	003216	100411					BMI 3\$;BRANCH IF YES	
1055	003220	012700	120000				MOV #120000,R0	;RESET ADDRESSING REGISTER	
1056	003224	062737	000200	172352			ADD #200,KIPARS	;MAP NEXT 4K BANK	
1057	003232	012737	000001	177572			MOV #1,MMRO	;CLEAR MEMORY MANAGEMENT ERRORS	
1058	003240	000002					RTI	;RETURN TO CONTINUE WRITING	
1059	003242	005037	177572		3\$:		CLR MMRO	;TURN OFF MEMORY MANAGEMENT	
1060	003246	062706	000004				ADD #4,SP	;CLEAN UP THE STACK	
1061	003252	000207					RTS PC	;RETURN TO CALLER	

```

1062          .SBTTL CHECK MEMORY FOR CORRECT DATA PATTERN
1063
1064 003254 004767 000404 CHECK: JSR PC,MAP ;SET UP MEMORY MANAGEMENT REGISTERS
1065 003260 016704 000374 MOV LIMIT,R4 ;GET BANK COUNT
1066 003264 016705 000372 MOV DATA,R5 ;GET DATA PATTERN
1067 003270 012737 003332 000250 MOV #2$,@#MMVEC ;SET UP FOR MEMORY MANAGEMENT ABORTS
1068 003276 012700 004470 MOV #END,RO ;ADDRESS OF WHERE DATA PATTERN STARTS
1069 003302 062700 120000 ADD #120000,RO ;USE KIPARS FOR ADDRESSING
1070 003306 012737 000001 177572 MOV #1,@#MMRO ;TURN ON MEM. MANAGEMENT
1071 003314 012001 1$: MOV (RO)+,R1 ;READ THE DATA
1072 003316 020501 CMP R5,R1 ;IS IT GOOD?
1073 003320 001775 BEQ 1$ ;BRANCH IF YES
1074 003322 000000 HALT ;BAD DATA IS IN R1
1075 003324 010560 177776 MOV R5,-2(RO) ;WRITE GOOD DATA
1076 003330 000771 BR 1$ ;GO READ THE NEXT WORD
1077 003332 005304 2$: DEC R4 ;LAST BANK COMPARED?
1078 003334 100411 BMI 3$ ;BRANCH IF YES
1079 003336 012700 120000 MOV #120000,RO ;RESET ADDRESSING REGISTER
1080 003342 062737 000200 172352 ADD #200,@#KIPARS ;MAP NEXT 4K BANK
1081 003350 012737 000001 177572 MOV #1,@#MMRO ;CLEAR MEMORY MANAGEMENT ERRORS
1082 003356 000002 RTI ;CONTINUE TESTING
1083 003360 005037 177572 3$: CLR @#MMRO ;TURN OFF MEM. MANAGEMENT
1084 003364 062706 000004 ADD #4,SP ;CLEAN UP THE STACK
1085 003370 000207 RTS PC ;RETURN FOR CALL
    
```

```

1086          .SBTTL  PARITY ERROR HANDLER
1087
1089 003372 005327  PARERR: DEC      (PC)+      ;FIRST TIME IN?
1099 003374 000001  PARFLG: .WORD    1
1090 003376 002002          BGE      2$      ;BRANCH IF YES
1091 003400 000000          1$:  HALT          ;NO--ANOTHER PARITY ERROR OCCURRED WHILE
1092 003402 000776          BR      1$      ;PROCESSING THE FIRST ONE.
1093 J03404 032767 000077 174330 2$:  BIT      #77,HIADRS ;MSB'S OF PARITY ERROR ADDRESS=0?
1094 003412 001006          BNE      4$      ;BRANCH IF NO
1095 003414 023727 177740 004470  LMP     @#LOADRS,#END ;LSB'S ABOVE THE PROGRAM?
1096 003422 101002          BHI      4$      ;BRANCH IF YES
1097 003424 000000          3$:  HALT          ;PARITY ERROR IS IN THE PROGRAM
1098 003426 000776          BR      3$
1099 003430 000000          4$:  HALT          ;DATA PARITY ERROR OCCURRED
1100 003432 005737 177744          TST     @#MEMERR    ;DID AN ABORT OCCUR?
1101 003436 100405          BMI      5$      ;BRANCH IF YES
1102 003440 162700 000002          SUB     #2,R0      ;BACKUP BY ONE WORD
1103 003444 012705 000002          MOV     #BIT01,R5
1104 003450 074500          XOR     R5,R0
1105 003452 010320          5$:  MOV     R3,(R0)+ ;REWRITE THE BAD DATA
1106 003454 013737 177744 177744  MOV     @#MEMERR,@#MEMERR ;CLEAR ERROR INDICATORS
1107 003462 012767 000001 177704  MOV     #1,PARFLG  ;INITIALIZE PARITY ERROR FLAG
1108 003470 000002          RTI          ;CONTINUE TESTING
    
```

```

1109
1110          .SBTTL POWER FAIL ROUTINE
1111
1112 003472 012767 177777 000156 POWDOWN: MOV      #-1,FLAG      ;FIRST INSTRUCTION FLAG
1113 003500 005067 000152          CLR      FLAG        ;NOW CLEAR IT
1114 003504 012777 003632 000132          MOV      #ILLUP,DUVEC ;IF TOO FAST
1115 003512 011667 000124          MOV      (SP),ERRAD   ;SET THE ERROR ADDRESS
1116 003516 022706 000440          CMP      #440,SP     ;YELLOW OR RED?
1117 003522 100402          BMI      IS         ;NO
1118 003524 012706 001100          MOV      #STACK,SP   ;SET EMERGENCY STACK
1119 003530 010046          IS:    MOV      R0,-(6)  ;PUT
1120 003532 010146          MOV      R1,-(6)     ;THE
1121 003534 010246          MOV      R2,-(6)     ;REGISTERS
1122 003536 010346          MOV      R3,-(6)     ;ON
1123 003540 010446          MOV      R4,-(6)     ;THE
1124 003542 010546          MOV      R5,-(6)     ;STACK
1125 003544 010667 000104          MOV      SP,SAVE6    ;SAVE THE STACK POINTER
1126 003550 016700 175324          MOV      FACTOR,R0   ;SET TIME FACTOR
1127 003554 077001          SOB      R0          ;NOW WAIT
1128 003556 012777 003566 000060          MOV      #POWUP,DUVEC ;RESET THE UP VECTOR
1129 003564 000000          HALT                ;WAIT FOR POWER DOWN
1130
1131 003566 012777 003636 000054 POWUP:  MOV      #ILLDOWN,DUVEC ;SET TOO FAST DOWN VECTOR
1132 003574 016706 000054          MOV      SAVE6,SP   ;RESET SP
1133 003600 016700 175274          MOV      FACTOR,R0  ;SET TIME FACTOR
1134 003604 077001          SOB      R0         ;WAIT
1135 003606 012605          MOV      (6)+,R5    ;TAKE
1136 003610 012604          MOV      (6)+,R4    ;THE
1137 003612 012603          MOV      (6)+,R3    ;REGISTERS
1138 003614 012602          MOV      (6)+,R2    ;FROM
1139 003616 012601          MOV      (6)+,R1    ;THE
1140 003620 012600          MOV      (6)+,R0    ;STACK
1141 003622 012777 003472 000020          MOV      #POWDOWN,DUVEC ;RESET THE DOWN VECTOR
1142 003630 000113          JMP      (R3)       ;JUMP INDIRECT TO R3
1143
1144 003632 000000          ILLUP: HALT        ;POWER UP BEFORE POWER DOWN COMPLETE
1145 003634 000776          BR      -2         ;LOCK UP THE HALT
1146
1147 003636 000000          ILLDOWN: HALT     ;POWERED DOWN BEFORE UP COMPLETE
1148 003640 000776          BR      -2         ;LOCK UP THE HALT
1149
1150 003642 000000          ERRAD:  0          ;RETURN ADDRESS FROM POWER FAIL
1151
1152 003644 000024 000026          UVEC:   24,26     ;UP ADDRESS PAIR
1153 003650 000024 000026          DVEC:   24,26     ;DOWN ADDRESS PAIR
1154 003654 000000          SAVE6:  0         ;SOME PLACE TO PUT THE SP
1155 003656 000000          FLAG:   0         ;1 INSTRUCTION DOWN FLAG
1156 003660 000000          LIMIT:  0         ;TOP OF MEMORY
1157 003662 052525          DATA:  52525     ;WHAT IS TO BE WRITTEN INTO MEMORY
    
```


1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230

```

:*****
.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.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOS    ;;CALL FOR TYPEOUT
*      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*      .BYTE   M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*$STYPON---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
1201 003766 017646 000000 000211 $TYPOS: MOV      2(SP),-(SP)      ;; PICKUP THE MODE
1202 003772 116667 000001 000211 MOV      1(SP), $OFILL    ;; LOAD ZERO FILL SWITCH
1203 004000 112667 000207 MOV      (SP)+, $OMODE+1  ;; NUMBER OF DIGITS TO TYPE
1204 004004 062716 000002 ADD      #2, (SP)        ;; ADJUST RETURN ADDRESS
1205 004010 000406 BR        $TYPON
1206 004012 112767 000001 000171 $TYPOC: MOV      #1, $OFILL    ;; SET THE ZERO FILL SWITCH
1207 004020 112767 000006 000165 MOV      #6, $OMODE+1    ;; SET FOR SIX(6) DIGITS
1208 004026 112767 000005 000154 $TYPON: MOV      #5, $OCNT    ;; SET THE ITERATION COUNT
1209 004034 010346 MOV      R3, -(SP)      ;; SAVE R3
1210 004036 010446 MOV      R4, -(SP)      ;; SAVE R4
1211 004040 010546 MOV      R5, -(SP)      ;; SAVE R5
1212 004042 116704 000145 MOV      $OMODE+1, R4    ;; GET THE NUMBER OF DIGITS TO TYPE
1213 004046 005404 NEG      R4
1214 004050 062704 000006 ADD      #6, R4          ;; SUBTRACT IT FOR MAX. ALLOWED
1215 004054 110467 000132 MOV      R4, $OMODE      ;; SAVE IT FOR USE
1216 004060 116704 000125 MOV      $OFILL, R4      ;; GET THE ZERO FILL SWITCH
1217 004064 016605 000012 MOV      12(SP), R5      ;; PICKUP THE INPUT NUMBER
1218 004070 005003 CLR      R3              ;; CLEAR THE OUTPUT WORD
1219 004072 006105 15:   ROL      R5          ;; ROTATE MSB INTO "C"
1220 004074 000404 BR        3$            ;; GO DO MSB
1221 004076 006105 25:   ROL      R5          ;; FORM THIS DIGIT
1222 004100 006105 ROL      R5
1223 004102 006105 ROL      R5
1224 004104 010503 MOV      R5, R3
1225 004106 006103 35:   ROL      R3          ;; GET LSB OF THIS DIGIT
1226 004110 105367 000076 DECB    $OMODE          ;; TYPE THIS DIGIT?
1227 004114 100016 BPL     7$            ;; BR IF NO
1228 004116 042703 177770 BIC     #177770, R3    ;; GET RID OF JUNK
1229 004122 001002 BNE     4$            ;; TEST FOR 0
1230 004124 005704 TST     R4            ;; SUPPRESS THIS 0?

```

```

1231 004126 001403          BEO      5$          ;; BR IF YES
1232 004130 005204          4$: INC      R4          ;; DON'T SUPPRESS ANYMORE C'S
1233 004132 052703 000060  BIS      #'0,R3       ;; MAKE THIS DIGIT ASCII
1234 004136 052703 000040  5$: BIS      #' ,R3       ;; MAKE ASCII IF NOT ALREADY
1235 004142 110367 000040  MOV      R3,8$        ;; SAVE FOR TYPING
1236 004146 104400 004206  TYPE     8$          ;; GO TYPE THIS DIGIT
1237 004152 105367 000032  7$: DECB   $OCNT      ;; COUNT BY 1
1238 004156 003347          BGT      2$          ;; BR IF MORE TO DO
1239 004160 002402          BLT      6$          ;; BR IF DONE
1240 004162 005204          INC      R4          ;; INSURE LAST DIGIT ISN'T A BLANK
1241 004164 000744          BR       2$          ;; GO DO THE LAST DIGIT
1242 004166 012605          6$: MOV     (SP)+,R5    ;; RESTORE R5
1243 004170 012604          MOV     (SP)+,R4    ;; RESTORE R4
1244 004172 012603          MOV     (SP)+,R3    ;; RESTORE R3
1245 004174 016666 000002 000004  MOV     2(SP),4(SP) ;; SET THE STACK FOR RETURNING
1246 004202 012616          MOV     (SP)+,(SP)
1247 004204 000002          RTI
1248 004206          000          ;; RETURN
1249 004207          000          ;; STORAGE FOR ASCII DIGIT
1250 004210          000          ;; TERMINATOR FOR TYPE ROUTINE
1251 004211          000          ;; OCTAL DIGIT COUNTER
1252 004212 000000          $OCNT: .BYTE 0      ;; ZERO FILL SWITCH
1253          $OFILL: .BYTE 0  ;; NUMBER OF DIGITS TO TYPE
1254          $OMCDE: .WORD 0
1255          ;;*****
1256          .SBTTL TYPE ROUTINE
1257          ;;*ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1258          ;;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1259          ;;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1260          ;;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1261          ;;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1262          ;;*
1263          ;;*CALL:
1264          ;;*1) USING A TRAP INSTRUCTION
1265          ;;* TYPE ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCII STRING
1266          ;;*OR
1267          ;;* TYPE
1268          ;;* MESADR
1269          ;;*
1270          ;;*2) USING A JSR INSTRUCTION
1271          ;;* MOV PS,-(SP)          ;; PUSH PROCESSOR STATUS WORD ON THE STACK
1272          ;;* JSR PC,$TYPE          ;; CALL TYPE ROUTINE
1273          ;;* MESADDR          ;; FIRST ADDRESS OF MESSAGE
1274
1275 004214 105767 000213  $TYPE: TSTB   $TPFLG    ;; IS THERE A TERMINAL?
1276 004220 100002          BPL      1$          ;; BR IF YES
1277 004222 000000          HALT
1278 004224 000407          BR       3$          ;; HALT HERE IF NO TERMINAL
1279 004226 010046          1$: MOV     R0,-(SP)   ;; LEAVE
1280 004230 017600 000002  MOV     2(SP),R0     ;; SAVE R0
1281 004234 112046          2$: MOV     (R0)+,-(SP) ;; GET ADDRESS OF ASCII STRING
1282 004236 001005          BNE      4$          ;; PUSH CHARACTER TO BE TYPED ONTO STACK
1283 004240 005726          TST     (SP)+        ;; BR IF IT ISN'T THE TERMINATOR
1284 004242 012600          MOV     (SP)+,R0    ;; IF TERMINATOR POP IT OFF THE STACK
1285 004244 062716 000002  3$: ADD     #2,(SP)   ;; RESTORE R0
1286 004250 000002          RTI          ;; ADJUST RETURN PC
1286          ;; RETURN

```



```

1343 004440 010046          $TRAP:  MOV    RO, -(SP)          ;; SAVE RO
1344 004442 016600 000002  MOV    2(SP),RO          ;; GET TRAP ADDRESS
1345 004446 005740          TST    -(RO)             ;; BACKUP BY 2
1346 004450 111000          MOVB   (RO),RO           ;; GET RIGHT BYTE OF TRAP
1347 004452 016000 004460  MOV    $TRPAD(RO),RO    ;; INDEX TO TABLE
1348 004456 000200          RTS    RO                ;; GO TO ROUTINE
    
```

.SBTTL TRAP TABLE

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

; ROUTINE
 ; -----

```

1358 004460          $TRPAD:
1359 004460 004214          $TYPE  ;; CALL=TYPE      TRAP+0(104400) TTY TYPEOUT ROUTINE
1360 004462 004012          $TYPOC ;; CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
1361 004464 003766          $TYPOS ;; CALL=TYPOS       TRAP+4(104404) TYPE OCTAL NUMBER (NO LEADING ZEROS)
1362 004466 004026          $TYPON ;; CALL=TYPON       TRAP+6(104406) TYPE OCTAL NUMBER (AS PER LAST CALL)
1363 004470 000000          END:    0
1364
1365          .END
    
```


KDPAR4=	172370	550#			
KDPAR5=	172372	551#			
KDPAR6=	172374	552#			
KDPAR7=	172376	553#			
KDPDR0=	172320	524#			
KDPDR1=	172322	525#			
KDPDR2=	172324	526#			
KDPDR3=	172326	527#			
KDPDR4=	172330	528#			
KDPDR5=	172332	529#			
KDPDR6=	172334	530#			
KDPDR7=	172336	531#			
KERSTK=	001100	263#			
KIPAR0=	172340	535#	1160*		
KIPAR1=	172342	536#			
KIPAR2=	172344	537#			
KIPAR3=	172346	538#			
KIPAR4=	172350	539#			
KIPAR5=	172352	540#	1056*	1080*	1162*
KIPAR6=	172354	541#	1164*		
KIPAR7=	172356	542#	1166*		
KIPDR0=	172300	513#	1161*		
KIPDR1=	172302	514#			
KIPDR2=	172304	515#			
KIPDR3=	172306	516#			
KIPDR4=	172310	517#			
KIPDR5=	172312	518#	1163*		
KIPDR6=	172314	519#	1165*		
KIPDR7=	172316	520#	1167*		
KSP	=%000006	297#			
LF	= 000012	277#	736	1319	1325
LIMIT	003660	709#	1045	1065	1156#
LOAD	003150	1004	1044#		
LOADRS=	177740	387#	1095		
MAINT =	177750	391#			
MAP	003664	984	1044	1064	1160#
MAPH0 =	170202	630#			
MAPH00=	170202	566#	630		
MAPH01=	170206	568#	632		
MAPH02=	170212	570#	634		
MAPH03=	170216	572#	636		
MAPH04=	170222	574#	638		
MAPH05=	170226	576#	640		
MAPH06=	170232	578#	642		
MAPH07=	170236	580#	644		
MAPH1 =	170206	632#			
MAPH10=	170242	582#			
MAPH11=	170246	584#			
MAPH12=	170252	586#			
MAPH13=	170256	588#			
MAPH14=	170262	590#			
MAPH15=	170266	592#			
MAPH16=	170272	594#			
MAPH17=	170276	596#			
MAPH2 =	170212	634#			
MAPH20=	170302	598#			

MAPH21=	170306	600#	
MAPH22=	170312	602#	
MAPH23=	170316	604#	
MAPH24=	170320	606#	
MAPH25=	170326	608#	
MAPH26=	170332	610#	
MAPH27=	170336	612#	
MAPH3 =	170216	636#	
MAPH30=	170342	614#	
MAPH31=	170346	616#	
MAPH32=	170352	618#	
MAPH33=	170356	620#	
MAPH34=	170362	622#	
MAPH35=	170366	624#	
MAPH36=	170372	626#	
MAPH37=	170376	628#	
MAPH4 =	170222	638#	
MAPH5 =	170226	640#	
MAPH6 =	170232	642#	
MAPH7 =	170236	644#	
MAPLO =	170200	629#	
MAPLO0=	170200	565#	629
MAPLO1=	170204	567#	631
MAPLO2=	170210	569#	633
MAPLO3=	170214	571#	635
MAPLO4=	170220	573#	637
MAPLO5=	170224	575#	639
MAPLO6=	170230	577#	641
MAPLO7=	170234	579#	643
MAPL1 =	170204	631#	
MAPL10=	170240	581#	
MAPL11=	170244	583#	
MAPL12=	170250	585#	
MAPL13=	170254	587#	
MAPL14=	170260	589#	
MAPL15=	170264	591#	
MAPL16=	170270	593#	
MAPL17=	170274	595#	
MAPL2 =	170210	633#	
MAPL20=	170300	597#	
MAPL21=	170304	599#	
MAPL22=	170310	601#	
MAPL23=	170314	603#	
MAPL24=	170320	605#	
MAPL25=	170324	607#	
MAPL26=	170330	609#	
MAPL27=	170334	611#	
MAPL3 =	170214	635#	
MAPL30=	170340	613#	
MAPL31=	170344	615#	
MAPL32=	170350	617#	
MAPL33=	170354	619#	
MAPL34=	170360	621#	
MAPL35=	170364	623#	
MAPL36=	170370	625#	
MAPL37=	170374	627#	

UDPDR2=	177624	438#																			
UDPDR3=	177626	439#																			
UDPDR4=	177630	440#																			
UDPDR5=	177632	441#																			
UDPDR6=	177634	442#																			
UDPDR7=	177636	443#																			
UIPAR0=	177640	447#																			
UIPAR1=	177642	448#																			
UIPAR2=	177644	449#																			
UIPAR3=	177646	450#																			
UIPAR4=	177650	451#																			
UIPAR5=	177652	452#																			
UIPAR6=	177654	453#																			
UIPAR7=	177656	454#																			
UIPDR0=	177600	425#																			
UIPDR1=	177602	426#																			
UIPDR2=	177604	427#																			
UIPDR3=	177606	428#																			
UIPDR4=	177610	429#																			
UIPDR5=	177612	430#																			
UIPDR6=	177614	431#																			
UIPDR7=	177616	432#																			
USESTK=	000600	265#																			
USP	=%000006	299#																			
UVEC	003644	834#																			
SCHARC	004420	1308	1114*	1128*	1152*																
SCMTAG=	***** U	693	1322*																		
SCRIF	004434	1292	696																		
SDOAGN	003142	1028	1331*																		
SENDAD	003132	684	1033	1039#																	
SENDCT	003116	697	715	1035#																	
SEOP	003072	1022#	1030#																		
SEOPCT	003110	697*	1027*	1031																	
SFILLC	004432	1295	1329#																		
SFILLS	004431	1328#																			
SGET42	003122	1032#																			
SHD	= 000000	248																			
SLF	004435	1332#																			
SNULL	004430	1297	1327*																		
SNWTST=	000001	741#	767#	787#	810#	824#	851#	866#	894#	909#	927#	943#	961#	977#							
		998#																			
SOCNT	004210	1208#	1237*	1250#																	
SOMODE	004212	1203#	1207*	1212	1215*	1226*	1252#														
SPASS	003146	696*	1024*	1025*	1041#																
SRDCHR=	***** U	1363																			
SRDDEC=	***** U	1363																			
SRDLIN=	***** U	1363																			
SRDOCT=	***** U	1363																			
SSAVRE=	***** U	1363																			
SSETUP=	000024	692#	694	696	698	715	1024														
SSTUP	= 177777	692#																			
SSVPC	= 000204	682#	687																		
SSWR	= 040000	238#	248	255	256	257	698	746	772	792	815	829	856	871							
		899	914	932	948	966	932	1002	1019	1024	1034	1040	1042								
STN	= 000017	238#	248	741	745	746#	763	767	771	772#	785	787	791	792#							
		805	810	814	815#	822	824	828	829#	847	851	855	856#	864							

	866	870	871#	890	894	898	899#	907	909	913	914#	923	927
	931	932#	941	943	947	948#	957	961	965	966#	975	977	981
	982#	995	998	1002#	1006	1012							
STPB	1314*	1326#											
STPFLG	1275	1330#											
STPS	1312	1325#											
STRAP	694	1343#											
STRP =	1350#	1360#	1361#	1362#	1363#								
STRPAD	1347	1358#											
STYP8# =	1363												
STYPDS =	1363												
STYPE	1275#	1350	1359										
STYPEC	1294	1301	1307	1312#	1313								
STYPEX	1318	1320	1323#										
STYPOC	1206#	1360											
STYPOH	1205	1208#	1362										
STYPOS	1201#	1361											
SSGET4 =	1034#												
SSTRP =	1349#	1360	1361	1362	1363								
SCFILL	1202*	1206*	1216	1251#									
	653#	657	659#	682	683#	685#	687#	688#	720#	732#	736#	751	756
	760	778	782	798	802	841	845	1041	1042	1127	1134	1145	1148
	1325	1326	1327	1328	1329	1330	1331	1332	1333#				

U

U

WAIT	748	774	794	833	876	.									
XOR	1104														
.ASCII	1331														
.ASCIZ	720	724	732	736	740	1332									
.BYTE	727	728	1248	1249	1250	1251	1327	1328	1329	1330					
.ENABL	238														
.END	1365														
.ENDC	243	256	257	258	647	662	663	685	687	692	693	694	696	698	715
	717	720	724	728	729	732	736	740	742	743	744	746	768	769	770
	772	788	789	790	792	811	812	813	815	825	826	827	829	852	853
	854	856	867	868	869	871	895	896	897	899	910	911	912	914	928
	929	930	932	944	945	946	948	962	963	964	966	978	979	980	982
	999	1000	1001	1002	1015	1018	1019	1021	1024	1028	1031	1032	1034	1040	1041
.EQUIV	1042	1176	1254	1335	1344	1347	1349	1359	1360	1361	1362	1363			
	266	267	269	290	291	292	293	294	295	296	297	298	299	300	329
	330	331	332	333	334	335	336	337	338	357	358	359	360	361	362
	363	364	365	366	418	419	420	421	629	630	631	632	633	634	635
	636	637	638	639	640	641	642	643	644						
.EVEN	720	724	732	736	740	1333									
.IF	239	256	257	258	647	657	662	684	686	692	693	694	696	698	714
	715	716	719	723	727	728	731	735	739	741	743	746	767	769	772
	787	789	792	810	812	815	824	826	829	851	853	856	866	868	871
	894	896	899	909	911	914	927	929	932	943	945	948	961	963	966
	977	979	982	998	1000	1002	1014	1018	1019	1020	1021	1023	1027	1030	1032
.IFF	1034	1040	1042	1175	1253	1334	1343	1347	1349	1350	1360	1361	1362	1363	
	256	257	663	695	686	693	715	717	727	728	741	742	743	744	746
	767	768	769	770	772	787	788	789	790	792	810	811	812	813	815
	824	825	826	827	829	851	852	853	854	856	866	867	868	869	871
	894	895	896	897	899	909	910	911	912	914	927	928	929	930	932
	943	944	945	946	948	961	962	963	964	966	977	978	979	980	982
	998	999	1000	1001	1002	1015	1021	1023	1028	1031	1040	1176	1254	1335	1344
.IFT	720	724	732	736	740										
.IFTF	720	724	732	736	740										
.IIF	238	243	248	255	256	657	694	696	715	726	1024	1041	1042	1325	1326
.IRP	1327	1328	1329	1330	1331	1332	1333	1359	1360	1361	1362				
	692	741	767	787	810	824	851	866	894	909	927	943	961	977	998
.LIST	1023														
	238	648	657	692	715	720	724	732	736	740	741	746	767	772	787
	792	810	815	824	829	851	856	866	871	894	899	909	914	927	932
	943	948	961	966	977	982	998	1002	1024	1034	1349	1350	1359	1360	1361
.MACRO	1362	1363													
.MCALL	257	258	1350												
.MLIST	238	648													
	238	648	657	692	715	720	724	732	736	740	741	746	767	772	787
	792	810	815	824	829	851	856	866	871	894	899	909	914	927	932
	943	948	961	966	977	982	998	1002	1024	1034	1349	1350	1359	1360	1361
.PAGE	1362	1363													
	650	688	741	787	824	866	909	943	977	1014	1042	1062	1086	1109	1158
.REM	1.75														
.REPT	1														
.SBTTL	657	259	364	395	409	558	651	658	664	741	767	787	810	824	851
	866	894	909	927	943	961	977	998	1016	1043	1062	1086	1110	1158	1177
.TITLE	1255	1336	1351												
.WORD	238														
	657	684	686	690	1027	1030	1041	1089	1252	1322	1325	1326			

F04

MAINDEC-11-DEKBA-A POP-11/70 POWER FAIL MACY11 27(732) 20-SEP-76 13:27 PAGE 46
DEKBA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*, DEKBA.SEQ/SOL/CRF/PAGNUM=DEKBA
RUN-TIME: 25 13 3 SECONDS
RUN-TIME RATIO: 162/42=3.8
CORE USED: 15K (29 PAGES)

