

DA11F

BUS WINDOW EXERCISER
MD-11-DZDAB-A

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

NOV 1976
digital
MADE IN U.S.A

This microfiche card contains a grid of frames. The frames are arranged in approximately 12 rows and 6 columns. Each frame contains a small, high-contrast image of a document page, likely a technical manual or exercise sheet. The text within these frames is too small to be legible. The right side of the card is mostly blank, with some faint, illegible markings.

801

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER

MACY11 27(732)

05-OCT-76

11:05 PAGE 2

.REM %

MAINDEC-11-DZDAB-A
DZDABA.P11

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZDAB-A-D
PRODUCT NAME:	BUS WINDOW EXERCISER
DATE CREATED:	1-MAY-73
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JOE STUBBLEBINE

COPYRIGHT (C) 1972
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS.

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
8.3	POWER FAIL
9.	PROGRAM DESCRIPTION

36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84

MAINDEC-11-DZDAB-A
DESCRIPTION

BUS WINDOW EXERCISER

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140

1. ABSTRACT

THIS PROGRAM EXERCISES THE BUS WINDOW USING AN I/O DEVICE OR EXECUTING CODE THROUGHOUT THE OTHER PROCESSOR'S MEMORY. IT IS ASYNCHRONOUS AND 3/4 DUPLEX IN ITS OPERATION. IF THE WINDOW IS ABOVE 32K, KT11C/D WILL BE USED TO REACH THE WINDOW.

2. REQUIREMENTS

2.1 EQUIPMENT

TWO PDP11 STANDARD COMPUTERS WITH A MINIMUM OF 8K OF MEMORY KT11C/D FOR MEMORY EXPANSION (OPTIONAL)

2.2 STORAGE

PROGRAM STORAGE - THE ROUTINES USE MEMORY 0 - 17776

2.3 PRELIMINARY PROGRAMS

MEMORY DIAGNOSTICS
DEVICE DIAGNOSTICS
MAINDEC-11-DZDAA-A

3. LOADING PROCEDURE

- 1) LOAD THE ABS LOADER IF NOT IN MEMORY
- 2) LOAD PROGRAM WITH ABS LOADER
- 3) LOAD OTHER PROCESSOR
 - A) IF WINDOW IS NOT UP, USE ABS LOADER ON OTHER PROCESSOR
 - OR
 - B) IF THE WINDOW IS UP:
 - 1) LOAD ADDRESS 510
 - 2) START
 - 3) LOAD ADDRESS #500 IN OTHER MACHINE (* = WINDOW ADDRESS)
 - 4) START
 - 5) BOTH PROCESSORS WILL SELF START IF SWITCH 7 IS CLEAR.

4. STARTING PROCEDURE

- 1) THERE ARE TWO WAYS OF STARTING THE PROGRAM. FIRST OF ALL THE PROGRAM CAN BE LOADED INTO EACH PROCESSOR BY USING

E01

NRINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER

MACY11 27(732) 05-OCT-76 11:05 PAGE 5

141
142

THE ABS LOADER. THE PROGRAMS CAN THEN BE STARTED AT THE
STARTING ADDRESS SPECIFIED IN SECTION 4.3 . THE OTHER

143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198

OPTION IS TO LOAD THE PROGRAM INTO ONE PROCESSOR USING THE ABS LOADER AND THE HAVE THE WINDOW TRANSFER THE PROGRAM TO THE OTHER PROCESSOR AS DESCRIBED IN SECTION 3 . TO USE THIS METHOD THE WINDOW MUST BE WORKING, IT MUST HAVE A SIZE OF AT LEAST 4K, AND THE WINDOW CANNOT BE LOCATED AT 32K OR ABOVE.

2) IF THE WINDOW IS USED TO LOAD THE PROGRAM INTO PROCESSOR "B" AND SWITCH 7 IS CLEAR, THE PROGRAMS WILL SELF START. PROCESSOR "A" WILL AUTOMATICALLY BECOME THE MASTER.

3) IF THE PROGRAM DOES NOT AUTOMATICALLY SELF START DUE TO LOADING PROGRAM MANUALLY INTO BOTH PROCESSORS OR SWITCH 7 BEING SET WHEN LOADING THRU THE WINDOW, THEN IT BECOMES NECESSARY TO START EACH PROCESSOR AT A DIFFERENT STARTING ADDRESS. ONE PROCESSOR MUST BE STARTED AT SA=200 AND THE OTHER PROCESSOR MUST BE STARTED FROM SA 210. BY DOING THIS A MASTER SLAVE CONFIGURATION IS ESTABLISHED TO PREVENT RACE CONDITIONS WHEN ASKING FOR THE WINDOW. AFTER THE INITIAL START, WHICH MODIFIES SOME CODE, ALL RESTARTS MUST BE DONE FROM SA 200 FOR BOTH PROCESSORS.

4.1 CONTROL SWITCH SETTINGS

SEE ! (ALL DOWN FOR WORST CASE TESTING)

4.2 STARTING ADDRESS

SEE 4.0.

4.3 PROGRAM AND/OR OPERATOR ACTION

- 1) START AT SA 200 FOR ONE PROCESSOR AND SA 210 FOR THE OTHER PROCESSOR OR SELF START AS IN SEC. 4.0
- 2) TYPE DEVICE (RF11, RK11, RP11, RC11, TC11, TM11, OR EXECUTE) AND RETURN.
- 3) TEST WILL START (↑C WILL RETURN TO STEP 2)
- 4) NOTE: ALL PROGRAM RESTARTS ARE FROM SA200 FOR BOTH PROCESSORS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

AT STARTING ADDRESS, ALL SWITCHES DOWN IS WORST CASE TESTING. THE BELL WILL RING UPON COMPLETION OF A PASS.

GO1

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER

MACY11 27(732) 05-OCT-76 11:05 PAGE 7

199

200
100
200
300
400
500
600
700
800
900
1000
1100
1200
1300
1400
1500
1600
1700
1800
1900
2000
2100
2200
2300
2400
2500
2600
2700
2800
2900
3000
3100
3200
3300
3400
3500
3600
3700
3800
3900
4000
4100
4200
4300
4400
4500
4600
4700
4800
4900
5000
5100
5200
5300
5400
5500
5600
5700
5800
5900
6000
6100
6200
6300
6400
6500
6600
6700
6800
6900
7000
7100
7200
7300
7400
7500
7600
7700
7800
7900
8000
8100
8200
8300
8400
8500
8600
8700
8800
8900
9000
9100
9200
9300
9400
9500
9600
9700
9800
9900
10000

5.1.1 SWITCH SETTINGS ARE:

SW<15> = 1 HALT ON ERROR
SW<14> = 1 HANG ON CURRENT SEGMENT
SW<13> = 1 INHIBIT PRINTOUT
SW<10> = 1 INHIBIT BELL ON PASS COMPLETE
SW<08> = 1 TYPE SEGMENT UNDER TEST
SW<07> = 1 INHIBIT AUTOMATIC SELF START WHEN LOADING
PROGRAM THRU THE WINDOW

5.2 SUBROUTINE ABSTRACTS

5.2.1 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 4 - 776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR + 2.

5.2.2 TRAP HANDLER

MOST OF THE SUBROUTINE CALLS ARE DONE VIA A TRAP+N CALL. TO FIND THE SUBROUTINE BEING CALLED, LOOK IN THE COMMENT SECTION OF THE TRAP DEFINITION TABLE FOR THE NAME OF THE TRAP, THEN SCAN TO THE LEFT MARGIN FOR THE STARTING ADDRESS OF THE SUBROUTINE.

6. ERRORS

6.1 ERROR PRINTOUT

THE FORMAT IS AS FOLLOWS:

RF11 ERROR: CS = 100305 ER = 001000

RF11 = DEVICE UNDER TEST
CS = CONTROL AND STATUS REGISTER
ER = ERROR REGISTER

DATA ERROR AT 401220 TRUE = 177777 RECEIVED = 177757

401220 = ADDRESS OF BAD DATA (18 BIT)
TRUE = DATA SENT
RECEIVED = DATA FOUND

WINDOW ERROR AT 146012 WCSRA = 141022 WDARA = 000000

146012 = ADDRESS TRAPPED FROM

256
257

WCSRA = WINDOW CONTROL AND STATUS REG.
WDARA = WINDOW DISPLACEMENT ADDRESS REGISTER

26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113

MAINDEC-11-DZDAB-A
DESCRIPTION

BUS WINDOW EXERCISER

PAGE 6

6.2 ERROR RECOVERY

RESTART AT 200 FOR BOTH PROCESSORS

7. RESTRICTIONS

IN ORDER TO ESTABLISH A MASTER/SLAVE CONFIGURATION, THE PROGRAMMER MUST INITIALLY BE STARTED AT TWO DIFFERENT STARTING ADDRESSES. IF THE WINDOW IS USED TO LOAD THE PROGRAM AND SWITCH 7 IS CLEAR, THE MASTER/SLAVE CONFIGURATION WILL BE SETUP. IF THE PROGRAMS ARE MANUALLY STARTED REFER TO SECTION 4.0 FOR STARTING INSTRUCTIONS. ALL RESTARTS ARE FROM SA 200 FOR BOTH PROCESSORS.

8. MISCELLANEOUS

WHEN FINISHED RUNNING THE PROGRAM THE LOADER MAY BE RESTORED BY LOADING SA 562 AND HITTING START. THE PROCESSOR WILL RESTORE THE LOADER AND HALT.

8.1 EXECUTION TIME

THE EXECUTION TIME IS DEPENDANT ON THE AMOUNT OF MEMORY AND THE DEVICE USED. THE BELL SHOULD RING WITHIN 5 MINUTES (USING TC11 IN 124K.)

8.2 STACK POINTER

STACK IS INITALLY SET TO 500

8.3 POWER FAIL

N/A

9. PROGRAM DESCRIPTION

THE PROGRAM SIZES CORE AND TEST SEQUENTIALLY ALL CORE OR ANY SECTION OF CORE IN CHUNKS EQUAL TO THE WINDOW SIZE USING THE DEVICE SPECIFIED. SO A SEGMENT OF CORE IS EQUAL TO THE WINDOW SIZE. THE DEVICE ON SIDE A USES THE MEMORY ON BUS B AND THE DEVICE ON SIDE B USES MEMORY ON BUS A. WORST CASE NOISE PATTERNS ARE USED FOR EACH MEMORY TYPE ON EACH PASS. IF A DEVICE IS NOT AVAILABLE, THE EXECUTE COMMAND CAN BE USED TO EXECUTE CODE THROUGHOUT THE OTHER PROCESSOR'S MEMORY. EACH MACHINE HAS A WINDOW INTERRUPT HANDLER FOR OPENING THE WINDOW UPON REQUEST AND CODE TO REQUEST THE

K01

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER

MACY11 27(732) 05-OCT-76 11:05 PAGE 11

314
315

WINDOW IN A PARTICULAR MODE. BECAUSE BOTH PROCESSORS CAN
NOT READ AT THE SAME TIME, THE REQUESTING ROUTINE WAITS FOR

MAINDEC-11-DZDAB-A
DESCRIPTION

BUS WINDOW EXERCISER

PAGE 7

THE OTHER PROCESSOR TO DROP THE READ REQUEST BEFORE
REQUESTING READ PRIVLAGES HIMSELF.

%
:TITLE MAINDEC-11-DZDAB-A BUS WINDOW EXERCISER
:COPYRIGHT 1972,DIGITAL EQUIPMENT CORP., MAYNARD MASS.
:PROGRAM BY BOB BRAIN

	SWITCH	USE
100000	SW15= 100000	:HALT ON ERROR
040000	SW14= 40000	:HANG ON CURRENT SEGMENT
020000	SW13= 20000	:INHIBIT ERROR TYPEOUTS
010000	SW12= 10000	
004000	SW11= 4000	
002000	SW10= 2000	:0 - BELL ON PASS COMPLETE
		:1 - BELL ON ERROR
001000	SW9= 1000	:INHIBIT USE OF MEMORY EXPANSION DEVICE
000400	SW8= 400	:TRACE BANK UNDER TEST
000200	SW7= 200	:HALT ON LOAD COMPLETE

BIT	WCSRA
15	A ERROR
14	B TIME OUT
13	B ACLO
12	B NEW DATA
11	B DATA 3
10	B DATA 2
9	B DATA 1
8	A TRANS ENABLE
7	B TRANS ENABLE
6	A I/E
5	A DATA 3
4	A DATA 2
3	A DATA 1
2	B WRITE ENABLE
1	A WRITE ENABLE
0	A NEW DATA

:FUNCTIONS (IN THE COMMUNICATION BITS)

	FUNCTION	MEANING
000001	CLEAR= 01	:NOP - CLOSE WINDOW
000011	READ= 11	:READ ONLY
000021	WRITE= 21	:WRITE ONLY
000031	R.W= 31	:READ/WRITE
000041	BANK= 41	:BANK SELECT

316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER

MACY11 27(732)

MO1

05-OCT-76 11:05 PAGE 13

372
373

⋮

51 & 61
71

⋮ FUTURE USE
⋮ GET READY FOR FUNCTION

374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429

104400
104000
000004
177776
177570
177570
000007
000000
000001
000002
000003
000004
000005
000005
000006
000007
000000
000000
000200
000200 000137 001026
000210 012737 000240 006660
000216 012737 000240 006662
000224 012737 000240 007106
000232 000137 001026
000500
000500 000167 000010
000504 000137 000562
000510 000137 011026
000514 010701
000516 000005
000520 162701 000516

;LOADING PROCEDURE:

- ;1) LOAD THE ABS LOADER IF NOT IN MEMORY
- ;2) LOAD PROGRAM WITH ABS LOADER
- ;3) LOAD ADDRESS 510
- ;4) START
- ;5) LOAD ADDRESS *500 IN OTHER MACHINE (* = WINDOW ADDRESS)
- ;6) START

SCOPE= TRAP
HLT= EMT
TYPE= IOT
PS= 177776
SWR= 177570
DISPLAY=SWR
BELL= 7
R0= %0
R1= %1
R2= %2
R3= %3
R4= %4
R5= %5
TTY= %5
SP= %6
PC= %7
N= 0

. = 0 ;TRAP CATCHER FROM 0 - 776

. = 200

JMP @#BEGIN ;JUMP TO BEGINING ADDRESS OF PROGRAM

;STARTING ADDRESS OF MASTER

. = 210
MOV #240,@#DOF2
MOV #240,@#DOF2+2
MOV #240,@#DOF5
JMP @#BEGIN

. = 500

JMP GETCOR ;GET THE PROGRAM
JMP @#LOADR ;RESTORE THE LOADER
JMP @#OPEN ;OPEN THE WINDOW

;THIS ROUTINE IS USED BY THE OTHER PROCESSOR TO LOAD THE PROGRAM
;INTO ITSELF. IT IS RUN AT LOAD TIME ONLY BY LOADING THE STARTING
;ADDRESS OF THE WINDOW PLUS 500 INTO THE SWITCHES. THEN DO A LOAD
;ADDRESS AND HIT START. THE OTHER PROCESSOR WILL THEN EXECUTE THE
;FOLLOWING CODE.

GETCOR: MOV PC,R1 ;GET WINDOW ADDRESS
RESET ;CLEAR THE WORLD
SUB @#GETCOR+2,R1 ;RELOCATE

430	000524	005000			CLR	RO		; SETUP FOR TRANSFER
431	000526	012120			MOV	(1)+,(0)+		; GET IT
432	000530	022700	017500		CMP	#17500,RO		; END?
433	000534	001374			BNE	1\$; NO - LOOP
434	000536	032737	000200	177570	BIT	#SW7,#SWR		; CHECK FOR HALT ON LOAD COMPLETE
435	000544	001401			BEQ	+.4		; CONTINUE
436	000546	000000			HALT			; WAIT FOR CONTINUE
437	000550	012767	000240	010306	MOV	#NOP,BR.		; MAKE IT EXECUTE A JMP #BEGIN
438	000556	000137	001026		JMP	#BEGIN		; START THE TEST

; IF YOU WISH TO RESTORE THE ABSOLUTE LOADER, LOAD THE ADDRESS OF
 ; THE FOLLOWING ROUTINE AND HIT START. THE PROCESSOR WILL HALT WHEN
 ; COMPLETE.

439	000562	004767	010714		LOADR:	JSR	PC,LORES	; RESTORE THE LOADER
440	000566	000000				HALT		


```

447      001000      . =      1000
448
449      001000      000000      ICNT:      0
450      001002      000000      ERRORS:     0
451      001004      000000      000000      PCNT:      0,0
452      001010      000000      LAD:      0
453      001012      000000      HLTADR:   0
454      001014      000000      MX:      0
455      001016      000000      MIN:      0
456      001020      000000      LIMIT:    0
457      001022      000000      OLIN:     0
458      001024      000000      OPTION:   0
459
460
461      001026      005037      000006      BEGIN:    CLR      286
462      001032      005067      010126      CLR      CFLAG
463      001036      012706      000500      MOV      #500,SP
464      001042      012737      011572      000020      MOV      #IOTS,2820
465      001050      012737      000340      000022      MOV      #340,2822
466      001056      012737      012346      000024      MOV      #PDOWNS,2824
467      001064      012737      000340      000026      MOV      #340,2826
468      001072      012737      006252      000034      MOV      #TRAPS,2834
469      001100      012737      000340      000036      MOV      #340,2836
470      001106      012777      000340      010264      MOV      #340,28CTRAP+2
471      001114      012777      000340      010252      MOV      #340,28FTRAP+2
472      001122      012777      000340      010254      MOV      #340,28PTRAP+2
473      001130      012777      000340      010252      MOV      #340,28KTRAP+2
474      001136      012777      000340      010250      MOV      #340,28TCTRAP+2
475      001144      012777      000340      010246      MOV      #340,28MTRAP+2
476      001152      012737      007614      000250      MOV      #MMERR,28250
477      001160      012737      000340      000252      MOV      #340,28252
478      001166      004767      006324      JSR      PC,MARK
479      001172      012767      011122      176660      MOV      #READR,60
480      001200      015067      177612      CLR      MIN
481      001204      012767      000000      177602      MOV      #0,MX
482      001212      005067      006226      CLR      MEMORY
483      001216      005067      010052      CLR      NOTRD
484      001222      005067      177576      CLR      OPTION
485      001226      022767      177777      010036      CMP      #-1,ONCE
486      001234      001402      BEQ      IS
487      001236      000167      001322      JMP      REDO
488      001242      032737      001000      177570      1S:      BIT      #SW9,28SWR
489      001250      001007      BNE      SIZEIT
490      001252      012737      001270      000004      MOV      #SIZEIT,284
491      001260      005767      176306      TST      SR0
492      001264      105167      177534      COMB     OPTION

```

```

: LH = ITERATION COUNT ; RH = TEST NO.
: ERROR COUNT
: 2 WORD PASS COUNT
: LOOP ADDRESS FOR SCOPE
: LAST HLT INSTRUCTION ADD. EXECUTED
: SEGMENT UNDER TEST
: FIRST BANK
: LAST BANK
: OTHER MACHINE LIMIT
: OPTION - GT = KT11

```

```

: HALT
: CLEAR CONTROL "C" FLAG
: SET STACK TO ** 500 **
: LOAD IOT VECTOR
: LOCK UP
: LOAD PF VECTOR
: LOCK UP
: SET FOR TRAP
: LOCK UP WORLD
: BR5
: BR5
: BR5
: BR5
: BR5
: BR5
: BR5
: BR5
: SET FOR MEMORY MANAGEMENT ERROR
: BR7
: ENABLE TRAP ON MEMORY PARITY ERRORS
: KEYBOARD VECTOR
: RESET MINIMUM BANK
: INIT
: WHICH MEMORY BEING CHECKED
: TYPE THE OPTION
: IS THIS FIRST TIME THRU?
: BRANCH IF YES
: INHIBIT USE OF MEMORY EXTENTION DEVICE
: SKIP
: SET FOR TIMEOUT
: CHECK FOR KT11
: OPTION IS KT11

```



```

493 001270 012737 000006 000004 SIZEIT: MOV #6,2#4 ;RESET FOR MEM
494 001276 005067 007766 CLR MMN ;CLEAR MEMORY MANAGEMENT FLAG
495 001302 005767 177516 TST OPTION ;WHICH OPTION AM I
496 001306 001016 BNE DOSEG ;MUST HAVE MEMORY MANAGEMENT
497
498
499
500 ;THIS ROUTINE IS USED TO SIZE CORE IF MEMORY MANAGEMENT IS NOT
501 ;AVAILABLE.
502
503 001310 012737 001342 000004 DOCORE: MOV #25,2#4 ;SET FOR MEM
504 001316 012701 017776 MOV #17776,R1 ;SET UP ADDRESS
505 001322 005000 CLR RO ;SET UP BANK COUNT
506 001324 062701 020000 1S: ADD #20000,R1 ;MOVE TO NEXT BANK
507 001330 005200 RO ;INC THE BANK COUNT
508 001332 005711 TST (1) ;TIMEOUT?
509 001334 022701 177776 CMP #177776,R1 ;END?
510 001340 001371 1S BNE 1S ;LOOP IF NOT AT THE END
511 001342 000432 2S: BR TYPEIT ;TYPE IT
512
513 ;THIS ROUTINE IS USED TO SIZE CORE IF MEMORY MANAGEMENT IS
514 ;AVAILABLE.
515
516
517 001344 104446 DOSEG: MAPIT ;SETUP MEMORY MANAGEMENT REGISTERS
518 001346 005267 176220 INC SRD ;TURN ON MEMORY MANAGEMENT
519 001352 012767 000200 170764 MOV #200,KIPAR2 ;INIT TO BANK 1
520 001360 012737 001410 000004 MOV #25,2#4 ;SET TIMEOUT ADDRESS FOR CORE CALCULATIONS
521 001366 005737 057776 1S: TST #57776 ;TRAP ON NON EX MEM
522 001372 062767 000200 170744 ADD #200,KIPAR2 ;GO TO NEXT BANK
523 001400 022767 007600 170736 CMP #7600,KIPAR2 ;LAST ONE?
524 001406 003367 BGT 1S ;TRY NEXT
525 001410 016700 170730 2S: MOV KIPAR2,RO ;SAVE PAR2 IN RO
526 001414 006300 ASL RO ;*ASK #-7,RO*
527 001416 000300 SWAB RO ;KLUDGY ISN'T IT
528 001420 042700 177740 BIC #177740,RO ;CLEAR JUNK
529 001424 005067 176142 CLR SRD ;TURN IT OFF
530
531
532 001430 005300 TYPEIT: DEC RO ;DROP BACK
533 001432 012737 000006 000004 MOV #6,2#4 ;SET FOR MEM
534 001440 012706 000500 MOV #500,SP ;CLEAR STACK **500**
535 001444 005267 007622 INC ONCE ;TYPE THE OPTION ONLY ONCE
536 001450 004767 007752 JSR 7,LODGET ;GET THE LOADER ONLY THE FIRST TIME
537 001524 010005 MOV RO,TTY ;TYPE RO IN OCTAL
538 001526 004767 010400 JSR PC,PRINTS ;AND SUPPRESS LEADING ZERO'S
539 001532 000004 001536 TYPE #+2 ;.ASCIZ "EXIST"
540 001546 005767 177252 TST OPTION ;WHICH OPTION?
541 001552 001410 BEQ NOKT ;GET OUT IF NONE
542 001554 000004 001560 TYPE #,+2 ;.ASCIZ " WITH KT11"

```


E02

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER MACY11 27(732) 05-OCT-76 11:05 PAGE 18
SETUP, CORE, AND OPTION SCAN

```

543 001574 010067 177220      NOKT:  MOV      RD,LIMIT      ;SAVE SIZE
544 001600 000004 001604      TYPE      ,.+2          ;.ASCIZ <15><12><12>
545 001610 012737 001624 000004 3S:  MOV      #7S,2#4      ;SET FOR TRAPPING
546 001616 005777 007414      TST      @WCSRA      ;IS THE ADDRESS LEAGAL
547 001622 000440      BR       FNDCSR      ;YES
548 001624 022626      CMP      (6)+,(6)+    ;CLEAR STACK
549 001626 012767 164000 007402 7S:  MOV      #164000,WCSRA ;SET WCSRA FOR SCANNING
550 001634 012701 000010      MOV      #8,R1        ;SET COUNT
551 001640 012737 001654 000004 8S:  MOV      #5S,2#4      ;SET FOR TIMEOUT
552 001646 005777 007364      TST      @WCSRA      ;IS IT THERE?
553 001652 000424      BR       FNDCSR      ;YES - GET OUT
554 001654 022626      CMP      (6)+,(6)+    ;CLEAR STACK
555 001656 062767 000020 007352 5S:  ADD      #20,WCSRA    ;BUMP ADDRESS
556 001664 005301      DEC      R1           ;DEC COUNT
557 001666 001367      BNE      6S          ;LOOP UNTIL FOUND
558 001670 000004 001674      TYPE      ,.+2          ;.ASCIZ "WCSRA NOT FOUND"<15><12><12>
559 001720 000000      HALT
560 001722 000732      BR       3S          ;TRY IT AGAIN

```

```

;THIS ROUTINE WILL DETERMINE WHERE THE WINDOW'S REGISTERS ARE AND
;THEN PROCEED TO TYPE OUT ITS FUNCTIONAL PARAMETERS. IT ALSO DETERMINES
;WETHER OR NOT MEMORY MANAGEMENT IS REQUIRED. IF SO, IT SETS UP THE
;NECESSARY MEMORY MANAGEMENT REGISTERS.

```

```

561
562
563
564
565
566
567
568
569 001724      FNDCSR:
570 001724 000004 001730      TYPE      ,.+2          ;.ASCIZ "WCSRA = "
571 001742 016705 007270      MOV      @WCSRA,TTY   ;TYPE WCSRA IN OCTAL
572 001746 004767 010150      JSR      PC,PRINTR    ;TYPE LEADING ZERO'S
573 001752 000004 001756      TYPE      ,.+2          ;.ASCIZ "
574 001762 000402      BR       32S         ;SKIP TRAP AREA
575 001764 000000      HALT          ;WINDOW ADDRESS TIMED OUT
576 001766 022626      CMP      (6)+,(6)+    ;CLEAR STACK
577 001770 012737 001764 000004 32S:  MOV      #31S,2#4     ;SET 4 FOR TIMEOUT
578 001776 016700 007234      MOV      @WCSRA,RD    ;GET THE FIRST ADD
579 002002 012701 011240      MOV      @WDBRA,R1    ;GET TABLE ADDRESS
580 002006 005720      TST      (0)+        ;ADD 2
581 002010 010021      MOV      RD,(1)+      ;GET THE DATA
582 002012 022701 011256      CMP      @WVECA,R1    ;END?
583 002016 001373      BNE      50S         ;LOOP UNTIL DONE
584 002020 012737 000006 000004 21S:  MOV      #6,2#4       ;RESET 4
585 002026 012777 177777 007212      MOV      #-1,@WRRARA  ;FIND THE WINDOW SIZE
586 002034 017700 007206      MOV      @WRRARA,RD   ;SAVE THE SIZE
587 002040 010067 007222      MOV      RD,SIZE      ;SAVE THE SIZE FOR POSTERITY
588 002044 005400      NEG      RD          ;BUMP IT TO 1 BIT ONLY
589 002046 000300      SWAB      RD         ;GET UPPER BITS
590 002050 006200      ASR      RD          ;MOVE IT INTO POSITION
591 002052 005001      CLR      R1         ;INIT FOR OFFSET
592 002054 005700      FND1:  TST      RD          ;IS IT 0
593 002056 001404      BEQ      43S        ;YES - GET OUT
594 002060 006200      ASR      RD          ;COUNT BITS
595 002062 062701 000002      ADD      #2,R1       ;KEEP COUNT
596 002066 000772      BR       FND1       ;LOOP
597 002070 016167 011212 007132 43S:  MOV      WIND(1),SIZ  ;GET THE SIZE
598 002076 010167 007132      MOV      R1,OFFSET   ;SAVE OFFSET

```


599	002102	000004	011230			TYPE	,SIZ	:TYPE THE WINDOW SIZE
600	002106	000004	002112			TYPE	+2	:ASCIZ "K WINDOW AT "
601	002130	017700	007114			MOV	2WADRAX,RO	:GET WINDOW ADDRESS
602	002134	005067	176654			CLR	MX	:GET READY TO TYPE
603	002140	012701	000005			MOV	#5,R1	:INIT COUNT
604	002144	006300		52S:		ASL	RO	:MOVE IT LEFT
605	002146	006167	176642			ROL	MX	:PICK UP DROPPED BIT
606	002152	005301				DEC	R1	:COUNT IT
607	002154	001373				BNE	52S	:LOOP UNTIL 0
608	002156	017700	007066			MOV	2WADRAX,RO	:GET AGAIN
609	002162	006300				ASL	RO	:GET FOR
610	002164	006300				ASL	RO	:TYPING
611	002166	010067	007070			MOV	RO, WADRA	:SAVE THE WINDOW ADDRESS
612	002172	010005				MOV	RO, TTY	:TYPE RO WITH MX AS 18 BIT ADDRESS
613	002174	004767	007756			JSR	PC, PRINTA	:GO TO ADDRESS PRINTER
614	002200	005067	007060			CLR	EMBITS	:CLEAR EXTENDED MEMORY BITS
615	002204	017700	007040			MOV	2WADRAX,RO	:GET WINDOW ADDRESS
616	002210	006300				ASL	RO	
617	002212	103003				BCC	1S	:BRANCH IF BIT 17 CLEAR
618	002214	052767	000040	007042		BIS	#40,EMBITS	:SET MEMORY EXTENDED BIT 5
619	002222	006300			1S:	ASL	RO	
620	002224	103003				BCC	2S	:BRANCH IF BIT 16 CLEAR
621	002226	052767	000020	007030		BIS	#20,EMBITS	:BRANCH IF BIT 16 CLEAR
622	002234	010067	007162		2S:	MOV	RO, 10WAD	
623	002240	012737	002402	000004		MOV	#FND2, 2#4	:SETUP TIMEOUT TRAP
624	002246	005737	177572			TST	2#SRD	:TEST FOR MM
625	002252	012767	177777	007010		MOV	#-1,MMN	:SET MEMORY MANAGEMENT FLAG
626	002260	005067	170054			CLR	KIPAR0	:MAP TO 0
627	002264	012767	077406	170006		MOV	#77406, KIPDR0	:READ/WRITE 4K
628	002272	012702	000005			MOV	#5, R2	:SET THE COUNT
629	002276	012701	172304			MOV	#KIPDR2, R1	:GET THE ADDRESS
630	002302	012721	077406		47S:	MOV	#77406, (1)+	:SET THE DPR
631	002306	005302				DEC	R2	:CHECK COUNT
632	002310	001374				BNE	47S	:LOOP
633	002312	012767	007600	170036		MOV	#7600, KIPAR7	:MAP TO BANK 37
634	002320	012767	077406	167770		MOV	#77406, KIPDR7	:READ/WRITE 4K
635	002326	017700	006716			MOV	2WADRAX, RO	:GET ADDRESS
636	002332	006000				ROR	RO	:MAKE
637	002334	006000				ROR	RO	:IT
638	002336	006000				ROR	RO	:PAGE
639	002340	006000				ROR	RO	:ADDRESS
640	002342	042700	170000			BIC	#170000, RO	:CLEAR JUNK
641	002346	010067	167772			MOV	RO, KIPAR2	:SET IT
642	002352	012702	000004			MOV	#4, R2	:SET THE COUNT
643	002356	012701	172346			MOV	#KIPAR3, R1	:GET ADDRESS
644	002362	062700	000200		57S:	ADD	#200, RO	:GO TO NEXT
645	002366	010021				MOV	RO, (1)+	:SET IN THE ADDRESS
646	002370	005302				DEC	R2	:COUNT
647	002372	001373				BNE	57S	:LOOP
648	002374	012767	040000	006660		MOV	#40000, WADRA	:FUDGE ADDRESS
649	002402	012737	000006	000004	FND2:	MOV	#6, 2#4	:RESET 4
650	002410	017700	006636			MOV	2WVARA, RO	:GET VECTOR ADDRESS
651	002414	000004	002420			TYPE	+2	:ASCIZ " VECTOR = "
652	002436	022700	001000			CMF	#1000, RO	:CHECK FOR UNDER 1000
653	002442	002403				BLT	25S	:GO TO ERROR ROUTINE
654	002444	022700	000136			CMF	#136, RO	:CHECK FOR OVER 136

655	002450	002402				BLT	265		:SKIP IF IN RANGE
656	002452	000000			255:	HALT			:VECTOR IS NOT BETWEEN 140-776
657	002454	000411				BR	FND3		:GET OUT IF CONTINUED
658	002456	010067	006574		265:	MOV	RO,WVECA		:SAVE THE VECTOR ADDRESS
659	002462	005720				TST	(0)+		:ADD 2
660	002464	010067	006570			MOV	RO,WVECA+2		:SET UP STATUS
661	002470	017705	006556			MOV	2WVARA,TTY		:TYPE 2WVARA IN OCTAL
662	002474	004767	007432			JSR	PC,PRINTS		:AND SUPPRESS LEADING ZERO'S
663	002500	012767	170000	006716	FND3:	MOV	8-4096.,WRDCNT		:SETUP MAXIMUM WORDCOUNT
664	002506	032767	000400	006552		BIT	8400,SIZE		:5K WINDOW?
665	002514	001404				BEQ	15		:BRANCH IF NO
666	002516	012767	177000	006700		MOV	8-512.,WRDCNT		:LIMIT WORDCOUNT TO 512
667	002524	000417				BR	REDO		
668	002526	032767	001000	006532	15:	BIT	81000,SIZE		:1K WINDOW?
669	002534	001404				BEQ	25		:BRANCH IF NO
670	002536	012767	176000	006660		MOV	8-1024.,WRDCNT		:LIMIT WORDCOUNT TO 1000
671	002544	000407				BR	REDO		
672	002546	032767	002000	006512	25:	BIT	82000,SIZE		:2K WINDOW?
673	002554	001403				BEQ	REDO		:BRANCH IF NO
674	002556	012767	174000	006640		MOV	8-2048.,WRDCNT		:LIMIT WORDCOUNT TO 2K
675	002564	012777	010404	006464	REDO:	MOV	8WINDOW,2WVECA		:SET WINDOW VECTOR
676	002572	005037	177776			CLR	2#PS		:LOWER PROCESSOR STATUS
677	002576	012777	000340	006454		MOV	8340,2WVECA+2		:HALT ADDRESS
678	002604	042777	100000	006424		BIC	810000,2WCSRA		:KILL ERROR
679	002612	052777	000100	006416		BIS	8100,2WCSRA		:I/E
680	002620	012737	000340	000006		MOV	8340,2#6		:BR7
681	002626	005767	006436			TST	MMN		:MEMORY MANAGEMENT NEEDED?
682	002632	001403				BEQ	15		:BRANCH IF NO
683	002634	012737	000001	177572		MOV	81,2#SR0		:TURN ON MEMORY MANAGEMENT
684	002642	032767	004000	006416	15:	BIT	84000,SIZE		:IS WINDOW OVER 4K?
685	002650	001006				BNE	TTYIN		:BRANCH IF NO
686	002652	042767	020000	006402		BIC	820000,WADRA		
687	002660	042767	020000	006534		BIC	820000,IOWAD		

:THE FOLLOWING CODE WILL DETERMINE WHICH TEST THE OPERATOR SELECTED
:AND TRANSFER CONTROL TO IT, IF THE COMMAND CANNOT BE INTERPRETED,
:IT WILL TYPE A LIST OF AVAILABLE COMMANDS.

```

688
689
690
691
692 002666 012737 010326 000004 TTYIN: MOV      #TRAP4,2#4      ;SET UP TRAP IF NO DEVICE
693 002674 000004 002700          TYPE      .+2          ;.ASCIZ <15><12><12>"*"
694 002706 004767 007034          JSR      PC,READS     ;INPUT DATA FROM TTY
695 002712 004767 006610          JSR      PC,CONTC    ;CHECK FOR RESTART
696 002716 012737 000340 177776 MOV      #340,2#PS    ;LOCK UP THE WORLD
697 002724 104450 000001          FUNCTN   CLEAR      ;REQUEST WINDOW IN CLEAR MODE
698 002730 005077 006304          CLR      2#DBRA     ;BANK 0
699 002734 104450 000041          FUNCTN   ,BANK      ;REQUEST WINDOW IN BANK MODE
700 002740 104450 000011          FUNCTN   ,READ      ;REQUEST WINDOW IN READ MODE
701 002744 016700 006312          MOV      2#ADRA,RO   ;GET WINDOW ADDRESS
702 002750 016067 001020 176044 MOV      LIMIT(0),OLIM ;GET THE SIZE
703 002756 032767 004000 006302 BIT      #4000,SIZE   ;4K WINDOW?
704 002764 001026          BNE      2$         ;BRANCH IF 4K OR LESS
705 002766 032767 010000 006272 BIT      #10000,SIZE  ;8K WINDOW?
706 002774 001403          BEQ      1$         ;BRANCH IF NO
707 002776 006267 176020          ASR      OLIM       ;CONVERT BANKS TO WINDOW SEGMENTS
708 003002 000417          BR       2$
709 003004 032767 020000 006254 1$: BIT      #20000,SIZE ;16K WINDOW?
710 003012 001405          BEQ      3$         ;BRANCH IF NO
711 003014 006267 176002          ASR      OLIM       ;CONVERT BANKS TO WINDOW SEGMENTS
712 003020 006267 175776          ASR      OLIM
713 003024 000406          BR       2$
714 003026 006267 175770 3$: ASR      OLIM       ;CONVERT BANKS TO WINDOW SEGMENTS
715 003032 006267 175764          ASR      OLIM
716 003036 006267 175760          ASR      OLIM
717 003042          BR       2$
718 003042 104450 000001          FUNCTN   ,CLEAR     ;REQUEST WINDOW IN CLEAR MODE
719 003046 012737 003466 000004 RTPF: MOV      #NODEV,2#4  ;SET UP TRAP IF NO DEVICE
720 003054 012706 000500          MOV      #500,SP     ;RESET THE STACK ** 500 **
721 003060 005037 177776          CLR      2#PS       ;DROP PRIORITY
722 003064 052737 000100 177560 BIS      #100,2#177560 ;IE THE TTY
723 003072 042777 100000 006136 BIC      #100000,2#CSRA ;KILL ERROR
724 003100 052777 000100 006130 BIS      #100,2#CSRA  ;I/E
725 003106 022767 041522 006736 CMP      #"RC,INPUT  ;CHECK FOR RC11
726 003114 001010          BNE      T1        ;TRY NEXT
727 003116 012777 004164 006252 MOV      #TRP.RC,2#CTRAP ;SET RC VECTOR
728 003124 052777 000100 006156 BIS      #100,2#CCS   ;TURN ON INTERRUPT
729 003132 000167 005154          JMP      WAIT       ;GO TO WAIT
730 003136 022767 050122 006706 T1: CMP      #"RP,INPUT  ;CHECK FOR RP11
731 003144 001010          BNE      T2        ;TRY NEXT
732 003146 012777 004624 006226 MOV      #TRP.RP,2#PTRAP ;SET RP VECTOR
733 003154 052777 000100 006140 BIS      #100,2#PCS   ;TURN ON INTERRUPT
734 003162 000167 005124          JMP      WAIT       ;GO TO WAIT
735 003166 022767 043122 006656 T2: CMP      #"RF,INPUT  ;CHECK FOR RF11
736 003174 001010          BNE      T3        ;TRY NEXT
737 003176 012777 003676 006166 MOV      #TRP.RF,2#FTRAP ;SET RF VECTOR
738 003204 052777 000100 006064 BIS      #100,2#FCS   ;TURN ON INTERRUPT
739 003212 000167 005074          JMP      WAIT       ;GO TO WAIT
740 003216 022767 045522 006626 T3: CMP      #"RK,INPUT  ;CHECK FOR RK11
741 003224 001010          BNE      T4        ;TRY NEXT
742 003226 012777 005004 006152 MOV      #TRP.RK,2#KTRAP ;SET RK VECTOR
743 003234 052777 000100 006074 BIS      #100,2#KCS   ;TURN ON INTERRUPT

```


MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
TTY INPUT AREA

MACY11 27(732) 05-OCT-76 11:05 PAGE 22

744	003242	000167	005044			JMP	WAIT	:GO TO WAIT
745	003246	022767	046524	006576	T4:	CMP	#TM, INPUT	:CHECK FOR TM11
746	003254	001010				BNE	T5	:TRY NEXT
747	003256	012777	005772	006132		MOV	#TRAP.TM, #TMTRAP	:SET TM VECTOR
748	003264	052777	000100	006070		BIS	#100, #TMCS	:TURN ON INTERRUPT
749	003272	000167	005014			JMP	WAIT	:GO TO WAIT
750	003276	022767	041524	006546	T5:	CMP	#TC, INPUT	:CHECK FOR TC11
751	003304	001013				BNE	T6	:TRY NEXT
752	003306	012777	005314	006076		MOV	#TRAP.TC, #TCTRAP	:SET TC VECTOR
753	003314	012767	000010	002436		MOV	#10, #TC.TA	:INITIALIZE STARTING BLOCK
754	003322	052777	000100	006020		BIS	#100, #TCCS	:TURN ON INTERRUPT
755	003330	000167	004756			JMP	WAIT	:GO TO WAIT
756	003334	022767	054105	006510	T6:	CMP	#EX, INPUT	:CHECK FOR EXECUTE CODE
757	003342	001005				BNE	T9	:GO TO NEXT
758	003344	012737	010326	000004		MOV	#TRAP4, #4	:RESET 4
759	003352	000167	000162			JMP	EXCODE	:EXECUTE THE CODE
760	003356				T8:			
761	003356	000004	003362			TYPE	.+2	:ASCIZ <15><12><12>"PC11, RP11, RF11, RK11, TM11, TC11,
762	003456	005037	177560			CLR	#177560	:ID THE TTY
763	003462	000167	177200			JMP	TTYIN	:LOOP AND TRY AGAIN

764
765
766
767
768
769
770
771
772
773

003466 005037 177776
003472 000004 003476
003502 000004 012052
003506 000004 003512
003534 000137 002564

; THIS ROUTINE IS ENTERED IF THE OPERATOR SELECTS A DEVICE WHICH
; DOES NOT EXIST.

NODEV: CLR @#PS
TYPE ,+2
TYPE ,INPUT
TYPE ,+2
JMP @#REDO

; LOWER PROCESSOR STATUS
;.ASCIZ <15><12><12>
; TYPE DEVICE NAME
;.ASCIZ " DOES NOT EXIST" <15><12>
; RETURN TO TTY INPUT MODE


```

774      ;THE FOLLOWING CODE PERFORMS THE EXECUTE COMMAND. IT WILL MOVE
775      ;THE LITTLE PROGRAM STARTING AT "CODE" THRU THE WINDOW TO THE OTHER
776      ;PROCESSOR AND THEN EXECUTE THE CODE IN THE OTHER PROCESSOR'S MEMORY.
777      ;IT WILL EXECUTE THE CODE IN ALL AVAILABLE BANKS OF THE OTHER PROCESSOR.
778
779

```

```

780 003540 012737 000340 177776 EXCODE: MOV      #340, @#PS      ;LOCK UP THE WORLD
791 003546 104444          NEXT          ;GET THE NEXT BANK
782 003550 104450 000031  FUNCTN   ,R.W        ;REQUEST WINDOW IN R.W MODE
783 003554 016700 005502  MOV      @ADRA,R0     ;GET ADDRESS OF WINDOW
784 003560 012701 003626  MOV      @CODE,R1     ;GET ADDRESS OF CODE
785 003564 012120          IS:    MOV      (1)+,(0)+    ;MOVE THE WORLD
786 003566 022701 003672  CMP      @ENDC+2,R1   ;END?
787 003572 001374          BNE      IS          ;LOOP UNTIL DONE
788 003574 016701 005462  MOV      @ADRA,R1     ;GET ADDRESS OF WINDOW
789 003600 062701 000043  ADD      @ENDC-CODE+1,R1 ;GET ENDC ADDRESS AND ODD
790 003604 004777 005452  JSR      PC,@ADRA     ;EXECUTE THE CODE
791 003610 104450 000001  FUNCTN   ,CLEAR      ;REQUEST WINDOW IN CLEAR MODE
792 003614 005037 177776  CLR      @#PS        ;DROP THE Prio TO 0
793 003620 004767 005702  JSR      PC,CONTC    ;CHECK FOR RESTART
794 003624 000745          BR       EXCODE     ;LOOP FOR EVER
795
796
797

```

```

798 003626 105421 CODE:  NEGB   (1)+      ;THIS
799 003630 106041      RORB  -(1)        ;CODE
800 003632 105421      NEGB   (1)+      ;IS
801 003634 106141      ROLB  -(1)        ;WORST
802 003636 106121      ROLB  (1)+       ;CASE
803 003640 105441      NEGB  -(1)        ;BUS
804 003642 106021      RORB  (1)+       ;NOISE
805 003644 105441      NEGB  -(1)        ;TO
806 003646 105421      NEGB  (1)+       ;CAUSE
807 003650 106041      RORB  -(1)        ;HAVIC
808 003652 105421      NEGB  (1)+       ;ON
809 003654 106141      ROLB  -(1)        ;THE
810 003656 106121      ROLB  (1)+       ;U
811 003660 105441      NEGB  -(1)        ;NI
812 003662 106021      RORB  (1)+       ;BUS
813 003664 105441      NEGB  -(1)        ;!!!!!!
814 003666 000207      RTS    PC         ;RETURN
815

```

```

816 003670 000000 ENDC:  0           ;DATA BUFFER AREA
817
818

```



```

819                                     ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
820                                     ;ISSUE A WRITE COMMAND.
821
822 003672 104410                       W.RF:  CHK.RF                       ;CHECK FOR ERRORS
823 003674 104430                       CKDATA                          ;CHECK CORE
824
825                                     ;ENTERED HERE INITIALLY BY THE COMMAND INTERPRETER.
826
827 003676 005077 005374                 TRP.RF: CLR      @RFCS                ;CLEAR INT ENB
828 003702 004767 005620                 JSR      PC,CONTC                   ;CHECK FOR RESTART
829 003706 104404                         CORE                                ;LOAD CORE WITH DATA
830 003710 012777 003746 005454         MOV      @C.RF,@RFTRAP              ;'CHECK' TRAP ADDRESS
831 003716 016777 005502 005356         MOV      @RDCNT,@RFWC               ;WORD COUNT
832 003724 005077 005356                 CLR      @RFDA                      ;DSK ADDRESS
833 003730 104450 000011                 FUNCTN  ,READ                       ;REQUEST WINDOW IN READ MODE
834 003734 104440                         .START                             ;LOAD CURRENT ADDRESS & IE!3 AND GO
835 003736 011304 000103 011276         @RCA,100!3,@RFS
836 003744 000002                         RTI
837
838                                     ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
839                                     ;ISSUE A WRITECHECK COMMAND.
840
841 003746 104410                       C.RF:  CHK.RF                       ;CHECK FOR ERRORS
842 003750 012777 004006 005414         MOV      @R.RF,@RFTRAP              ;'READ' TRAP ADDRESS
843 003756 016777 005442 005316         MOV      @RDCNT,@RFWC               ;WORD COUNT
844 003764 005077 005316                 CLR      @RFDA                      ;DSK ADDRESS
845 003770 104450 000011                 FUNCTN  ,READ                       ;REQUEST WINDOW IN READ MODE
846 003774 104440                         .START                             ;LOAD CURRENT ADDRESS & IE!7 AND GO
847 003776 011304 000107 011276         @RCA,100!7,@RFS
848 004004 000002                         RTI
849
850                                     ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL
851                                     ;THEN ISSUE A READ COMMAND.
852
853 004006 104410                       R.RF:  CHK.RF                       ;CHECK FOR ERRORS
854 004010 104406                       COMCOR                               ;COMPLIMENT CORE
855 004012 012777 003672 005352         MOV      @W.RF,@RFTRAP              ;'WRITE' TRAP ADDRESS
856 004020 016777 005400 005254         MOV      @RDCNT,@RFWC               ;WORD COUNT
857 004026 005077 005254                 CLR      @RFDA                      ;DSK ADDRESS
858 004032 104450 000021                 FUNCTN  ,WRITE                       ;REQUEST WINDOW IN WRITE MODE
859 004036 104440                         .START                             ;LOAD CURRENT ADDRESS & IE!5 AND GO
860 004040 011304 000105 011276         @RCA,100!5,@RFS
861 004046 000002                         RTI
862
863                                     ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
864
865 004050 005067 005114                 CH.RF: CLR      EX                   ;CLEAR ERROR FLAG
866 004054 104450 000001                 FUNCTN  ,CLEAR                       ;REQUEST WINDOW IN CLEAR MODE
867 004060 005777 005212                 TST     @RFCS                       ;ANY ERRORS?
868 004064 100034                         BPL     IS                           ;BRANCH IF NO ERRORS
869 004066 005267 005076                 INC     EX                           ;SET ERROR FLAG
870 004072 032737 020000 177570         BIT     @SW13,@SWR                  ;INHIBIT ERROR TYPEOUT
871 004100 001026                         BNE     IS                           ;INHIBIT TYPEOUTS
872 004102 017705 005170                 MOV     @RFCS,TTY                   ;SET TTY FOR TYPING
873 004106 000004 004112                 TYPE   ,+2                          ;ASCIZ <15><12>"RF11 ERROR:"
874 004130 000004 011172                 TYPE   ,M3                          ;TYPE CS

```


875	004134	004767	005762
876	004140	000004	011202
877	004144	017705	005130
878	004150	004767	005746
879	004154	104434	
880	004156	000002	

1S:

JSR	PC, PRINTR
TYPE	M4
MOV	ORFER, TTY
JSR	PC, PRINTR
STOP	
RTI	

:	TYPE	RFCS IN OCTAL
:	TYPE	ER
:	TYPE	ORFER IN OCTAL
:	TYPE	LEADING ZERO'S
:	HALT	ON ERROR
:	RETURN	


```

881 ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
882 ;ISSUE A WRITE COMMAND.
883
884 004160 104412 W.RC: CHK.RC ;CHECK FOR ERRORS
885 004162 104430 CKDATA ;CHECK CORE
886
887 ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETE.
888
889 004164 005077 005120 TRP.RC: CLR @RCCS ;CLEAR INT ENB
890 004170 004767 005337 JSR PC,CONTC ;CHECK FOR RESTART
891 004174 104404 CORE ;LOAD CORE WITH DATA
892 004176 012777 004734 005172 MOV #C.RC,@RCTRAP ;'CHECK' TRAP ADDRESS
893 004204 016777 005214 005102 MOV WRDCNT,@RCWC ;WORD COUNT
894 004212 005077 005102 CLR @RCDA ;DSK ADDRESS
895 004216 104450 000011 FUNCTN ,READ ;REQUEST WINDOW IN READ MODE
896 004222 104440 .START ;LOAD CURRENT ADDRESS & IE!3 AND GO
897 004224 011316 000103 011310 RCCA,100!3,RCCS
898 004232 000002 RTI
899
900 ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
901 ;ISSUE A WRITECHECK COMMAND.
902
903 004234 104412 C.RC: CHK.RC ;CHECK FOR ERRORS
904 004236 012777 004274 005132 MOV #R.RC,@RCTRAP ;'READ' TRAP ADDRESS
905 004244 016777 005154 005042 MOV WRDCNT,@RCWC ;WORD COUNT
906 004252 005077 005042 CLR @RCDA ;DSK ADDRESS
907 004256 104450 000011 FUNCTN ,READ ;REQUEST WINDOW IN READ MODE
908 004262 104440 .START ;LOAD CURRENT ADDRESS & IE!7 AND GO
909 004264 011316 000107 011310 RCCA,100!7,RCCS
910 004272 000002 RTI
911
912 ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL THEN
913 ;ISSUE A READ COMMAND.
914
915 004274 104412 R.RC: CHK.RC ;CHECK FOR ERRORS
916 004276 104406 COMCOR ;COMPLIMENT CORE
917 004300 012777 004160 005070 MOV #W.RC,@RCTRAP ;'WRITE' TRAP ADDRESS
918 004306 016777 005112 005030 MOV WRDCNT,@RCWC ;WORD COUNT
919 004314 005077 005000 CLR @RCDA ;DSK ADDRESS
920 004320 104450 000021 FUNCTN ,WRITE ;REQUEST WINDOW IN WRITE MODE
921 004324 104440 .START ;LOAD CURRENT ADDRESS & IE!5 AND GO
922 004326 011316 000105 011310 RCCA,100!5,RCCS
923 004334 000002 RTI
924
925 ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
926
927 004336 005067 004626 CH.RC: CLR EX ;CLEAR ERROR FLAG
928 004342 104450 000001 FUNCTN ,CLEAR ;REQUEST WINDOW IN CLEAR MODE
929 004346 005777 004736 TST @RCCS ;ANY ERRORS?
930 004352 100034 BPL IS ;BRANCH IF NO ERRORS
931 004354 015267 004610 INC EX ;SET ERROR FLAG
932 004360 012737 020000 177570 BIT #SW13,@SWR ;INHIBIT ERROR TYPEOUT
933 004366 001026 BNE IS ;INHIBIT TYPEOUTS
934 004370 011705 004714 MOV @RCCS,TTY ;SET TTY FOR TYPING
935 004374 000304 004400 TYPE ,+2 ;ASCIZ <15><12>"RC11 ERROR:"
936 004416 000004 011172 TYPE ,M3 ;TYPE CS

```


MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
ERROR ROUTINE FOR RC11

MACY11 27(732) 05-OCT-76 11:05 PAGE 28

937	004422	004767	005474
938	004426	000004	011202
939	004432	017705	004654
940	004436	004767	005460
941	004442	104434	
942	004444	000002	

IS:

JSR	PC,PRINTR
TYPE	M4
MOV	ORCER,TTY
JSR	PC,PRINTR
STOP	
RTI	

:	TYPE	RCCS IN OCTAL
:	TYPE	ER
:	TYPE	ORCER IN OCTAL
:	TYPE	LEADING ZERO'S
:	HALT	ON ERROR
:	RETURN	


```

943                                     ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
944                                     ;ISSUE A WRITE COMMAND.
945
946 004446 104414 W.RP: CHK.RP          ;CHECK FOR ERRORS
947 004450 104430      CKDATA          ;CHECK CORE
948 004452 005077 004644 TP.RP: CLR      ;RPCS          ;CLEAR INT ENB
949 004456 004767 005044      JSR      PC,CONTC ;CHECK FOR RESTART
950 004462 104404      CORE          ;LOAD CORE WITH DATA
951 004464 012777 004522 004710 MOV      ;C.RP ;RPTRAP ;'CHECK' TRAP ADDRESS
952 004472 016777 004726 004624 MOV      ;WORDCNT,;RPMC ;WORD COUNT
953 004500 005077 004624      CLR      ;RPOA          ;DSK ADDRESS
954 004504 104450 000011      FUNCTN  ,READ ;REQUEST WINDOW IN READ MODE
955 004510 104440      .START          ;LOAD CURRENT ADDRESS & IE!3 AND GO
956 004512 011326 000103 011322 RPCA,100!3,RPCS
957 004520 000002      RTI
958
959                                     ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
960                                     ;ISSUE A WRITECHECK COMMAND.
961
962 004522 104414 C.RP: CHK.RP          ;CHECK FOR ERRORS
963 004524 012777 004562 004650 MOV      ;R.RP ;RPTRAP ;'READ' TRAP ADDRESS
964 004532 016777 004666 004564 MOV      ;WORDCNT,;RPMC ;WORD COUNT
965 004540 005077 004564      CLR      ;RPOA          ;DSK ADDRESS
966 004544 104450 000011      FUNCTN  ,READ ;REQUEST WINDOW IN READ MODE
967 004550 104440      .START          ;LOAD CURRENT ADDRESS & IE!7 AND GO
968 004552 011326 000107 011322 RPCA,100!7,RPCS
969 004560 000002      RTI
970
971                                     ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL THEN
972                                     ;ISSUE A READ COMMAND.
973
974 004562 104414 R.RP: CHK.RP          ;CHECK FOR ERRORS
975 004564 104406      CONCOR          ;COMPLIMENT CORE
976 004566 012777 004446 004606 MOV      ;W.RP ;RPTRAP ;'WRITE' TRAP ADDRESS
977 004574 016777 004624 004522 MOV      ;WORDCNT,;RPMC ;WORD COUNT
978 004602 005077 004522      CLR      ;RPOA          ;DSK ADDRESS
979 004606 104450 000021      FUNCTN  ,WRITE ;REQUEST WINDOW IN WRITE MODE
980 004612 104440      .START          ;LOAD CURRENT ADDRESS & IE!5 AND GO
981 004614 011326 000105 011322 RPCA,100!5,RPCS
982 004622 000002      RTI
983
984                                     ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.
985
986 004624 012777 000011 004470 TRP.RP: MOV      #11,;RPCS ;INITIALIZE
987 004632 105777 004464      TSTB     ;RPCS ;ROUTINE
988 004636 100375      BPL      .-4 ;FOR
989 004640 005777 004470      TST     ;RPOA ;THE
990 004644 100375      BPL      .-4 ;RP11
991 004646 005077 004450      CLR     ;RPCS ;DISK
992 004652 000677      BR      TP.RP ;PACK

```



```

993                                     ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
994
995 004654 005067 004310                CH.RP: CLR      EX      ;CLEAR ERROR FLAG
996 004660 104450 000001                FUNCTN  CLEAR  ;REQUEST WINDOW IN CLEAR MODE
997 004664 005777 004432                TST     @RPCS  ;ANY ERRORS?
998 004670 100042                        BPL     IS     ;BRANCH IF NO ERRORS
999 004672 005267 004272                INC     EX     ;SET ERROR FLAG
1000 004676 032737 020000 177570        BIT     @SW13,@SWR ;INHIBIT ERROR TYPEOUT
1001 004704 001034                        BNE     IS     ;INHIBIT TYPEOUTS
1002 004706 017705 004410                MOV     @RPCS,TTY ;SET TTY FOR TYPING
1003 004712 000004 004716                TYPE   ',.+2    ;ASCIZ <15><12>"RP11 ERROR:"
1004 004734 000004 011172                TYPE   'M3      ;TYPE CS
1005 004740 004767 005156                JSR    PC,PRINTR ;TYPE RPCS IN OCTAL
1006 004744 000004 011202                TYPE   'M4      ;TYPE ER
1007 004750 017705 004356                MOV     @RPER,TTY ;TYPE @RPER IN OCTAL
1008 004754 004767 005142                JSR    PC,PRINTR ;TYPE LEADING ZERO'S
1009 004760 104434                        STOP    ;HALT ON ERROR
1010 004762 012777 000001 004332        MOV     #1,@RPCS ;LOAD A CLEAR
1011 004770 105777 004326                TSTB   @RPCS   ;WAIT FOR
1012 004774 100375                        BPL     .-4     ;DONE
1013 004776 000002                        RTI                                     ;RETURN

```



```

1014                                     ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
1015                                     ;ISSUE A WRITE COMMAND.
1016
1017 005000 104416                         W.RK:  CHK.RK                               ;CHECK FOR ERRORS
1018 005002 104430                         CKDATA                               ;CHECK CORE
1019
1020                                     ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.
1021
1022 005004 005077 004326                   TRP.RK: CLR      @RKCS                       ;CLEAR INT ENB
1023 005010 004767 004512                   JSR      PC,CONTC                          ;CHECK FOR RESTART
1024                                     CORE                                  ;LOAD CORE WITH DATA
1025 005014 104404                                     MOV      @C.RK,@RKTRAP                    ;CHECK TRAP ADDRESS
1026 005016 012777 005054 004362           MOV      @RDCNT,@RKWC                      ;WORD COUNT
1027 005024 016777 004374 004310           CLR      @RKDA                             ;DSK ADDRESS
1028 005032 005077 004310                   FUNCTN   ,READ                             ;REQUEST WINDOW IN READ MODE
1029 005036 104450 000011                   .START                                     ;LOAD CURRENT ADDRESS & IE!3 AND GO
1030 005042 104440                                     RKCA,100!3,RKCS
1031 005044 011344 000103 011336           RTI
1032
1033                                     ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL
1034                                     ;THEN ISSUE A WRITE CHECK COMMAND.
1035
1036 005054 104416                         C.RK:  CHK.RK                               ;CHECK FOR ERRORS
1037 005056 012777 005114 004322           MOV      @R.RK,@RKTRAP                    ;'READ' TRAP ADDRESS
1038 005064 016777 004334 004250           MOV      @RDCNT,@RKWC                      ;WORD COUNT
1039 005072 005077 004250                   CLR      @RKDA                             ;DSK ADDRESS
1040 005076 104450 000011                   FUNCTN   ,READ                             ;REQUEST WINDOW IN READ MODE
1041 005102 104440                                     .START                                     ;LOAD CURRENT ADDRESS & IE!7 AND GO
1042 005104 011344 000107 011336           RKCA,100!7,RKCS
1043 005112 000002                                     RTI
1044
1045                                     ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL THEN
1046                                     ;ISSUE A READ COMMAND.
1047
1048 005114 104416                         R.RK:  CHK.RK                               ;CHECK FOR ERRORS
1049 005116 104406                         COMCOR                                  ;COMPLIMENT CORE
1050 005120 012777 005000 004260           MOV      @W.RK,@RKTRAP                    ;'WRITE' TRAP ADDRESS
1051 005126 016777 004272 004206           MOV      @RDCNT,@RKWC                      ;WORD COUNT
1052 005134 005077 004206                   CLR      @RKDA                             ;DSK ADDRESS
1053 005140 104450 000021                   FUNCTN   ,WRITE                            ;REQUEST WINDOW IN WRITE MODE
1054 005144 104440                                     .START                                     ;LOAD CURRENT ADDRESS & IE!5 AND GO
1055 005146 011344 000105 011336           RKCA,100!5,RKCS
1056 005154 000002                                     RTI
1057
1058                                     ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
1059
1060 005156 005067 004006                   CH.RK: CLR      EX                          ;CLEAR ERROR FLAG
1061 005162 104450 000001                   FUNCTN   CLEAR                            ;REQUEST WINDOW IN CLEAR MODE
1062 005166 015777 004144                   TST     @RKCS                              ;ANY ERRORS?
1063 005172 100042                                     BPL     IS                                ;BRANCH IF NO ERRORS
1064 005174 005267 003770                   INC     EX                                  ;SET ERROR FLAG
1065 005200 032737 020000 177570           BIT     @SW13,@SWR                        ;INHIBIT ERROR TYPEOUT
1066 005206 001034                                     BNE     IS                                ;INHIBIT TYPEOUTS
1067 005210 017705 004122                   MOV     @RKCS,TTY                          ;SET TTY FOR TYPING
1068 005214 000004 005220                                     TYPE    ,+2                               ;ASCIZ <15><12>"RK11 ERROR:"
1069 005236 000004 011172                                     TYPE    ,M3                               ;TYPE CS

```


F03

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
ERROR ROUTINE FOR RK11

MACY11 27(732) 05-OCT-76 11:05 PAGE 32

1070	015242	004767	004654	JSR	PC,PRINTR	:TYPE RKCS IN OCTAL
1071	015246	000004	011202	TYPE	M4	:TYPE ER
1072	015252	017705	004062	MOV	MARKER,TTY	:TYPE MARKER IN OCTAL
1073	015256	004767	004640	JSR	PC,PRINTR	:TYPE LEADING ZERO'S
1074	015262	104434		STOP		:HALT ON ERROR
1075	015264	012777	000001 004044	MOV	#1,ARKCS	:LOAD A CLEAR
1076	015272	105777	004040	TSTB	ARKCS	:WAIT FOR
1077	015276	100375		BPL	.-4	:DONE
1078	015300	000002		RTI		:RETURN

15:


```

1079                                     ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
1080                                     ;ISSUE A WRITE COMMAND.
1081
1082 005302 104420 W.TC: CHK.TC ;CHECK FOR ERRORS
1083 005304 104430 CKDATA ;CHECK CORE
1084 005306 012777 000001 004034 MOV #1,@TCCS ;STOP ALL TRANSPORTS
1085
1086                                     ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.
1087
1088 005314 004767 004206 TRP.TC: JSR PC,CONTC ;CHECK FOR RESTART
1089 005320 062767 000001 000432 ADD #1,TC.TA ;UPDATE BLOCK
1090 005326 022767 000400 000424 CMP #400,TC.TA
1091 005334 003003 BGT IS
1092 005336 012767 000010 000414 MOV #10,TC.TA
1093 005344 104404 IS: CORE ;LOAD WORST CASE NOISE
1094 005346 104450 000011 FUNCTN ,READ ;REQUEST WINDOW IN READ MODE
1095 005352 104424 000010 SEARCH ,10 ;SEARCH FOR BLOCK
1096 005356 012777 005404 004026 MOV #R.TC,@TCTRAP ;TRAP ADDRESS
1097 005364 016777 004034 003762 MOV #RDCNT,@TCWC ;WORD COUNT
1098 005372 104440 .START
1099 005374 011356 000115 011350 TCCA,115,TCCS
1100 005402 000002 RTI
1101
1102                                     ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
1103                                     ;ISSUE A READ COMMAND.
1104
1105 005404 104420 R.TC: CHK.TC ;CHECK FOR ERROR
1106 005406 104406 COMCOR ;COMPLIMENT CORE
1107 005410 104450 000021 FUNCTN ,WRITE ;REQUEST WINDOW IN WRITE MODE
1108 005414 104424 000010 SEARCH ,10 ;SEARCH FOR BLOCK
1109 005420 012777 005302 003764 MOV #W.TC,@TCTRAP ;TRAP ADDRESS
1110 005426 016777 003772 003720 MOV #RDCNT,@TCWC ;WORD COUNT
1111 005434 104440 .START
1112 005436 011356 000105 011350 TCCA,105,TCCS
1113 005444 000002 RTI
1114
1115                                     ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
1116
1117 005446 005067 003516 CH.TC: CLR EX ;CLEAR ERROR FLAG
1118 005452 104450 000001 FUNCTN ,CLEAR ;REQUEST WINDOW IN CLEAR MODE
1119 005456 005777 003666 TST @TCCS ;ANY ERRORS?
1120 005462 100042 BPL IS ;BRANCH IF NO ERRORS
1121 005464 005267 003500 INC EX ;SET ERROR FLAG
1122 005470 032737 020000 177570 BIT #SW13,@SWR ;INHIBIT ERROR TYPEOUT
1123 005476 001034 BNE IS ;INHIBIT TYPEOUTS
1124 005500 017705 003644 MOV @TCCS,TTY ;SET TTY FOR TYPING
1125 005504 000004 005510 TYPE ,+2 ;ASCIZ <15><12>"TC11 ERROR:"
1126 005526 000004 011172 TYPE ,M3 ;TYPE CS
1127 005532 004767 004364 JSR PC,PRINTR ;TYPE TCCS IN OCTAL
1128 005536 000004 011202 TYPE ,M4 ;TYPE ER
1129 005542 017705 003604 MOV @TCER,TTY ;TYPE @TCER IN OCTAL
1130 005546 004767 004350 JSR PC,PRINTR ;TYPE LEADING ZERO'S
1131 005552 104434 STOP ;HALT ON ERROR
1132 005554 012777 000001 003566 MOV #1,@TCCS ;LOAD A CLEAR
1133 005562 105777 003562 TSTB @TCCS ;WAIT FOR
1134 005566 100375 BPL .-4 ;DONE

```


MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
ERROR ROUTINE FOR TC11

MACY11 27(732)

H03
05-OCT-76 11:05 PAGE 34

1135 005570 000002

15: RTI

;RETURN


```

1136                                     ;THE FOLLOWING ROUTINE WILL SEARCH FOR THE DESIGNATED
1137                                     ;DEC-TAPE BLOCK. THE DESIRED BLOCK IS STORED IN "TC.TA".
1138
1139 005572 062716 000002 S.TC: ADD #2,(6)
1140 005576 012777 005636 003606 MOV #SC1,@TCCTRAP ;TRAP ADDRESS
1141 005604 016767 000150 000144 MOV TC.TA,WANT
1142 005612 162767 000002 000136 SUB #2,WANT ;2 PAST BLOCK
1143 005620 012667 000140 MOV (6)+,RETURN ;SAVE RETURN ADDRESS
1144 005624 005726 TST (6)+
1145 005626 012777 004103 003514 RETI1: MOV #4103,@TCCS ;SEARCH REVERSE
1146 005634 000002 RTI
1147
1148 005636 005777 003506 SC1: TST @TCCS ;CHECK FOR ERROR
1149 005642 100010 BPL SC1A ;NO ERRORS
1150 005644 005777 003502 TST @TCER ;CHECK FOR END OF TAPE
1151 005650 100404 BMI IS
1152 005652 104436 ERR3
1153 005654 005767 003304 TST CFLAG ;IS IC TYPED?
1154 005660 001034 BNE SCQ1 ;BRANCH IF TYPED
1155 005662 000405 IS: BR SCF1
1156
1157 005664 026777 000066 003466 SC1A: CMP WANT,@TCDB ;ON BLOCK?
1158 005672 003001 BGT SCF1 ;GO REVERSE
1159 005674 000754 BR RETI1 ;CONTINUE AS YOU WERE
1160
1161 005676 012777 005714 003506 SCF1: MOV #SCF2,@TCCTRAP ;TRAP ADDRESS
1162 005704 012777 000103 003436 RETR1: MOV #103,@TCCS ;SEARCH FOWARD
1163 005712 000002 RTI
1164
1165 005714 005777 003430 SCF2: TST @TCCS ;ERROR?
1166 005720 100010 BPL SCF1A ;NO ERRORS
1167 005722 005777 003424 TST @TCER ;END OF TAPE?
1168 005726 100404 BMI IS
1169 005730 104432 ERR2
1170 005732 005767 003226 TST CFLAG ;NO
1171 005736 001005 BNE SCQ1 ;IS IC TYPED?
1172 005740 000732 IS: BR RETI1 ;BRANCH IF SET
1173 ;SWITCH DIRECTIONS
1174 005742 026777 000012 003410 SCF1A: CMP TC.TA,@TCDB ;RIGHT BLOCK?
1175 005750 001355 BNE RETR1 ;NO
1176 005752 000177 000006 SCQ1: JMP @RETURN
1177
1178 005756 000000 WANT: 0
1179 005760 000000 TC.TA: 0
1180 005762 000000 TC.WC: 0
1181 005764 000000 RETURN: 0

```



```

1182                                     ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
1183                                     ;ISSUE A WRITE COMMAND.
1184
1185 005766 104422                         W.TM:  CHK.TM                               ;CHECK FOR ERROR
1186 005770 104430                         CKDATA
1187
1188                                     ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.
1189
1190 005772 005077 003364                   TRP.TM: CLR      @TMCS                       ;CLEAR INT ENB
1191 005776 004767 003524                   JSR      PC,CONTC                          ;CHECK FOR RESTART
1192 006002 104404                           CORE                                       ;SET UP CORE
1193 006004 104426                           REWIND
1194 006006 012777 006074 003402             MOV      #R.TM,@TMTRAP                    ;TRAP ADDRESS
1195 006014 016777 003404 003344             MOV      WRDCNT,@TMBC                     ;BYTE COUNT
1196 006022 006377 003340                   ASL      @TMBC                             ;MAKE BYTE COUNT
1197 006026 104450 000011                   FUNCTN  ,READ                             ;REQUEST WINDOW IN READ MODE
1198 006032 104440                           .START                                    ;LOAD CURRENT ADDRESS & IE!60005 AND GO
1199 006034 011370 060105 011362             TMCA,100!60005,TMCS
1200 006042 000002                           RTI
1201
1202                                     ;REWIND THE TAPE
1203
1204 006044 012777 006070 003344             RW.TM:  MOV      #X.TM,@TMTRAP            ;TRAP ADDRESS
1205 006052 012667 000062                   MOV      (6)+,TMRET                       ;SAVE ADDRESS
1206 006056 005726                           TST      (6)+                              ;RESTORE STACK
1207 006060 012777 060117 003274             MOV      #60117,@TMCS                    ;REWIND
1208 006066 000002                           RTI
1209
1210 006070 000177 000044                   X.TM:   JMP      @TMRET
1211
1212                                     ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
1213                                     ;ISSUE A READ COMMAND.
1214
1215 006074 104422                         R.TM:   CHK.TM                               ;ANY ERRORS?
1216 006076 104406                         CONCOR                                       ;COMPLIMENT CORE
1217 006100 104426                           REWIND
1218 006102 012777 005766 003306             MOV      #W.TM,@TMTRAP                    ;TRAP ADDRESS
1219 006110 016777 003310 003250             MOV      WRDCNT,@TMBC                     ;BYTE COUNT
1220 006116 006377 003244                   ASL      @TMBC                             ;MAKE BYTE COUNT
1221 006122 104450 000021                   FUNCTN  ,WRITE                             ;REQUEST WINDOW IN WRITE MODE
1222 006126 104440                           .START                                    ;LOAD CURRENT ADDRESS & IE!60003 AND GO
1223 006130 011370 060103 011362             TMCA,100!60003,TMCS
1224 006136 000002                           RTI
1225
1226 006140 000000                         TMRET:  0
1227
1228                                     ;COME HERE TO CHECK THE STATUS OF THE PREVIOUS OPERATION.
1229
1230 006142 005067 003022                   CH.TM:  CLR      EX                          ;CLEAR ERROR FLAG
1231 006146 104450 000001                   FUNCTN  ,CLEAR                             ;REQUEST WINDOW IN CLEAR MODE
1232 006152 005777 003204                   TST      @TMCS                             ;ANY ERRORS?
1233 006156 100034                           BPL     IS                                 ;BRANCH IF NO ERRORS
1234 006160 005267 003004                   INC      EX                                 ;SET ERROR FLAG
1235 006164 032737 020000 177570             BIT      #SW13,@SWR                       ;INHIBIT ERROR TYPEOUT
1236 006172 001026                           BNE     IS                                 ;INHIBIT TYPEOUTS
1237 006174 017705 003162                   MOV      @TMCS,TTY                         ;SET TTY FOR TYPING

```


K03

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
ERROR ROUTINE FOR TM11

MACY11 27(732) 05-OCT-76 11:05 PAGE 37

1238	006200	000004	006204
1239	006222	000004	011172
1240	006226	004767	003670
1241	006232	000004	011202
1242	006236	017705	003122
1243	006242	004767	003654
1244	006246	104434	
1245	006250	000002	

15:

TYPE	,+2
TYPE	M3
JSR	PC,PRINTR
TYPE	M4
MOV	JTMR,TTY
JSR	PC,PRINTR
STOP	
RTI	

:	ASCIZ <15><12>"TM11 ERROR:"
:	TYPE CS
:	TYPE TMCS IN OCTAL
:	TYPE ER
:	TYPE JTMR IN OCTAL
:	TYPE LEADING ZERO'S
:	HALT ON ERROR
:	RETURN


```

1246
1247
1248
1249
1250
1251
1252
1253 006252 011605
1254 006254 162705 000002
1255 006260 111505
1256 006262 062705 006304
1257 006266 022705 006354
1258 006272 100401
1259 006274 000175 000000
1260 006300 000000
1261 006302 000776
1262
1263 006304
1264 006304 012506
1265 006306 012526
1266 006310 007446
1267 006312 007724
1268 006314 004050
1269 006316 004336
1270 006320 004654
1271 006322 005156
1272 006324 005446
1273 006326 006142
1274 006330 005572
1275 006332 006044
1276 006334 007140
1277 006336 005446
1278 006340 007336
1279 006342 007754
1280 006344 007354
1281 006346 010122
1282 006350 006356
1283 006352 010244
1284 006354 006602
1285 006354 006354

```

```

; STRAP ;TRAP HANDLER
; THIS ROUTINE DECODES A TRAP CALL AND JUMPS TO THE APROPRATE
; SUBROUTINE. THE CALL IS A "TRAP+N" WHERE N IS A MULTIPLE OF 2.
; THE "SET" MACRO WILL CREATE THE TABLE NEEDED. IT HAS TO
; FOLLOW THIS MACRO.

```

```

TRAPS:  MOV      (6),RS      ;GET TRAP INSTRUCTION
        SUB      #2,RS      ;BACK UP BY 2
        MOV      (5),RS      ;GET THE RIGHT BYTE OF TRAP
        ADD      #ADRTAB,RS  ;INDEX TO TABLE
        CMP      #ENDTAB,RS  ;CHECK FOR END OF TABLE
        BMI      .+4         ;OUT OF BOUNDS
        JMP      @2(5)       ;GO TO ROUTINE
        HALT
        BR       .-2        ;HANG UP

```

```

ADRTAB:  PATCH1      ;DUM1      = TRAP+0      (104400)
        PATCH2     ;DUM2      = TRAP+2      (104402)
        WRADR      ;CORE      = TRAP+4      (104404)
        WRCADR     ;COMCOR    = TRAP+6      (104406)
        CH.RF      ;CHK.RF    = TRAP+10     (104410)
        CH.RC      ;CHK.RC    = TRAP+12     (104412)
        CH.RP      ;CHK.RP    = TRAP+14     (104414)
        CH.RK      ;CHK.RK    = TRAP+16     (104416)
        CH.TC      ;CHK.TC    = TRAP+20     (104420)
        CH.TM      ;CHK.TM    = TRAP+22     (104422)
        S.TC       ;SEARCH    = TRAP+24     (104424)
        RW.TM      ;REWIND    = TRAP+26     (104426)
        CDATA      ;CKDATA    = TRAP+30     (104430)
        CH.TC      ;ERR2      = TRAP+32     (104432)
        POTS       ;STOP      = TRAP+34     (104434)
        ER3        ;ERR3      = TRAP+36     (104436)
        .BEGIN     ;.START    = TRAP+40     (104440)
        SETBAK     ;SETBAN    = TRAP+42     (104442)
        NXT        ;NEXT      = TRAP+44     (104444)
        MAP        ;MAPIT     = TRAP+46     (104446)
        DOFUN      ;FUNCTN    = TRAP+50     (104450)
ENDTAB=-2

```



```

1286                                     ; THIS ROUTINE WILL DETERMINE WHICH WINDOW SEGMENT WILL BE TESTED
1287                                     ; NEXT. LOCATION "OLIM" CONTAINS THE NUMBER OF WINDOW SEGMENTS
1288                                     ; AVAILABLE IN THE OTHER PROCESSOR. WHEN EXITED, LOCATION "MX"
1289                                     ; WILL CONTAIN THE NUMBER OF THE SEGMENT TO BE TESTED.
1290
1291 006356 032737 040000 177570 NXT:   BIT      #SW14, @#SWR      ; LOOP ON BANK?
1292 006364 001104                               BNE      2$          ; SKIP IF LOOP
1293 006366 026767 172422 172426       CMP      MX, OLIM    ; LAST PAGE?
1294 006374 002453                               BLT      1$          ; SKIP IF NOT LAST PAGE
1295 006376 032737 002000 177570       BIT      #SW10, @#SWR ; INHIBIT BELL
1296 006404 001002                               BNE      5$          ; YES!
1297 006406 000004 000007                               TYPE     , BELL      ; RING THE BELL
1298 006412 016767 172400 172374 5$:   MOV      MIN, MX     ; RESET MX
1299 006420 032767 004000 002640       BIT      #4000, SIZE ; 4K WINDOW OR LESS?
1300 006426 001402                               BEQ      8$          ; BRANCH IF GREATER
1301 006430 005267 172360                               INC      MX
1302 006434 016737 172354 177570 8$:   MOV      MX, @#DISPLAY ; DISPLAY IT
1303 006442 032737 000400 177570       BIT      #SW8, @#SWR ; TYPE BANK NUMBER??
1304 006450 001410                               BEQ      6$          ; NO!
1305 006452 000004 006456                               TYPE     , +2        ; .ASCIZ <15><12>" "
1306 006462 016705 172326                               MOV      MX, TTY     ; TYPE MX IN OCTAL
1307 006466 004767 003440                               JSR      PC, PRINTS  ; AND SUPPRESS LEADING ZERO'S
1308 006472 005767 000746 6$:       TST      MEMORY     ; CHECK STATE OF FLAG
1309 006476 100407                               BMI      4$          ; SWAB IT
1310 006500 001403                               BEQ      3$          ; COM IT
1311 006502 005067 000736                               CLR      MEMORY     ; INIT IT
1312 006506 000433                               BR       2$          ; CONTINUE
1313 006510 105167 000731 3$:       COMB    MEMORY+1   ; MAKE IT 177400
1314 006514 000430                               BR       2$          ; SKIP
1315 006516 000367 000722 4$:       SWAB    MEMORY     ; MAKE IT 377
1316 006522 000425                               BR       2$
1317 006524 005267 172264 1$:       INC      MX          ; GO TO NEXT BANK
1318 006530 016737 172260 177570     MOV      MX, @#DISPLAY ; DISPLAY THE BANK IN USE
1319 006536 032737 000400 177570     BIT      #SW8, @#SWR ; TYPE BANK NUMBER??
1320 006544 001414                               BEQ      2$          ; NO!
1321 006546 000004 006552                               TYPE     , +2        ; .ASCIZ " "
1322 006554 016767 172234                               MOV      MX, TTY     ; SET FOR TYPING
1323 006560 032705 000007                               BIT      #7, TTY     ; MULTIPLE OF 10?
1324 006564 001402                               BEQ      7$          ; YES
1325 006566 042705 177770 7$:       BIC      #177770, TTY ; CLEAR OUT UPPER BITS
1326 006572 004767 003334 7$:       JSR      PC, PRINTS ; TYPE MX AND SUPPRESS LEADING ZERES
1327 006576 104442 2$:       SETBANK ; SET MXC
1328 006600 000002                               RTI

```



```

1329 ; THIS ROUTINE WILL REQUEST THE OTHER PROCESSOR TO SETUP THE WINDOW
1330 ; IN RESPONSE TO A CODE IN THE COMMUNICATION BITS OF THE CSR, THE
1331 ; RESPONSE OF THE OTHER WINDOW IS THEN VERIFIED; AND IF IT IS WRONG
1332 ; AN ERROR IS REPORTED. OPERATIONS THROUGH THE WINDOW ARE RESTRICTED
1333 ; SO THAT EACH PROCESSOR CAN'T HAVE READ PRIVELGES AT THE SAME TIME.
1334 ; ONE OF THE PROCESSORS IS CONSIDERED A SLAVE TO THE OTHER. THIS IS
1335 ; NECESSARY TO AVOID RACE CONDITIONS WHEN REQUESTING THE WINDOW.
1336 ; WHEN THE SLAVE PROCESSOR IS IN THIS ROUTINE IT WILL LOWER ITS
1337 ; PRIORITY LEVEL TO 0 TO GIVE THE OTHER PROCESSOR THE RIGHT OF WAY.
1338 ; WHEN THE MASTER PROCESSOR IS IN THIS ROUTINE, HIS PRIORITY LEVEL
1339 ; IS KEPT AT 7. THE STARTING PROCEDURE DETERMINES WHICH PROCESSOR
1340 ; IS THE MASTER.
1341
1342 006602 005037 177776 DOFUN: CLR      2#PS      ; DROP PRIORITY
1343 006606 005767 002462      TST      NOTRD    ; IS THE READ FLAG SET?
1344 006612 001410      BEQ      3$      ; DO IT IF NO READ
1345 006614 032776 000010 000000 BIT      #10,2(6) ; IS IT A READ REQUEST?
1346 006622 001404      BEQ      3$      ; NO - DO IT
1347 006624 032777 000400 002404 2$: BIT      #400,2WCSRA ; R/W?
1348 006632 001374      BNE      2$      ; HANG UNTIL DONE
1349 006634 012737 000340 177776 3$: MOV      #340,2#PS ; LOCK UP THE WORLD
1350 006642 152777 000071 002366 BISB    #71,2WCSRA ; SEND FUNCTION GET READY
1351 006650 032777 000301 002360 DOF1: BIT      #1,2WCSRA ; WAIT FOR NEW DATA TO GO AWAY
1352 006656 001403      BEQ      DOF7    ; BRANCH IF GONE
1353 006660 005037 177776 DOF2: CLR      2#PS      ; LOWER STATUS IF SLAVE
1354 006664 000771      BR       DOF1    ; WAIT
1355 006666 142777 000070 002342 DOF7: BICB    #70,2WCSRA ; CLEAR THE FUNCTION BITS
1356 006674 157677 000000 002334 BISB    2(6),2WCSRA ; SET NEW FUNCTION BITS
1357 006702 032777 000001 002326 1$: BIT      #1,2WCSRA ; WAIT FOR NEW DATA
1358 006710 001374      BNE      1$      ; HANG
1359 006712 017604 000000      MOV      2(6),R4 ; GET THE FUNCTION BITS
1360 006716 010467 000214      MOV      R4,SAV1? ; REMEMBER OLD FUNCTION
1361 006722 006204      ASR      R4      ; SHIFT
1362 006724 006204      ASR      R4      ; TWICE
1363 006726 017705 002304      MOV      2WCSRA,R5 ; GET WCSRA
1364 006732 042705 177573      BIC      #17573,R5 ; CLEAR JUNK
1365 006736 026405 007116      CMP      CSRTAB(R4),R5 ; RIGHT RESPONSE?
1366 006742 001452      BEQ      DOF3    ; SKIP IF OK
1367 006744 012737 000340 177776 MOV      #340,2#PS
1368 006752 010401      MOV      R4,R1
1369 006754 010500      MOV      R5,R0
1370 006756 000004 006762      TYPE    #12 ; .ASCIZ <15><12><12>"FUNCTION IS WRONG: SENT = "
1371 007022 006301      ASL      R1
1372 007024 000261      SEC
1373 007026 006101      ROL      R1
1374 007030 010105      MOV      R1,TTY
1375 007032 004767 003074      JSR      PC,PRINTS ; AND SUPRESS LEADING ZERO'S
1376 007036 000004 007042      TYPE    #12 ; .ASCIZ " WCSRA = "
1377 007056 017705 002154      MOV      2WCSRA,TTY ; TYPE 2WCSRA IN OCTAL
1378 007062 004767 003034      JSR      PC,PRINTR ; TYPE LEADING ZERO'S
1379 007066 000000      HALT    ; FUNCTION NOT RIGHT!
1380 007070 005767 002200 DOF3: TST      NOTRD    ; IS THE MASTER READING?
1381 007074 001405      BEQ      DOF4    ; BRANCH IF NO
1382 007076 032776 000010 000000 BIT      #10,2(6) ; IS SLAVE TRYING TO READ?
1383 007104 001401      BEQ      DOF4    ; BRANCH IF NO
1384 007106 000770 DOF5: BR       DOF3    ; WAIT IF SLAVE

```


1385	007110	062716	000002	DOF4:	ADD	#2,(6)	:UPDATE THE RETURN ADDRESS
1386	007114	000002			RTI		:RETURN
1387							
1388	007116	000000		CSRTAB:	0		:CLEAR
1389	007120	000200			200		:READ
1390	007122	000204			204		:WRITE
1391	007124	000204			204		:READ/WRITE
1392	007126	000000	000000 000000		0,0,0		:N/A
1393	007134	000000			0		:GET READY
1394							
1395	007136	000000		SAVTMP:	0		


```

1396
1397
1398
1399
1400 007140 005767 002024
1401 007144 001401
1402 007146 000002
1403 007150 104450 000031
1404 007154 005067 000260
1405 007160 005067 000256
1406 007164 016702 002072
1407 007170 016705 002230
1408 007174 004767 000032
1409 007200 026712 000234
1410 007204 001401
1411 007206 104436
1412 007210 005722
1413 007212 005767 001746
1414 007216 001002
1415 007220 005205
1416 007222 001364
1417 007224
1418 007224 104450 000001
1419 007230 000002
1420
1421
1422
1423 007232 005767 000206
1424 007236 100430
1425 007240 003016
1426
1427 007242 010203
1428 007244 000303
1429 007246 006103
1430 007250 060203
1431 007252 006003
1432 007254 066703 000162
1433 007260 006003
1434 007262 103404
1435 007264 005167 000152
1436 007270 005167 000144
1437 007274 000207
1438 007276 010203
1439 007300 010201
1440 007302 000303
1441 007304 006101
1442 007306 006101
1443 007310 006101
1444 007312 006101
1445 007314 060103
1446 007316 000756
1447 007320 010203
1448 007322 006003
1449 007324 010301
1450 007326 000301
1451 007330 006003

```

```

;THIS ROUTINE WILL COMPARE THE DATA READ FROM A DEVICE AGAINST THE
;KNOWN PATTERN. ANY ERRORS ARE REPORTED AND THE ROUTINE WILL CONTINUE
;TO SCAN THE BUFFER.

```

```

CDATA: TST      EX
        BEQ     .+4
        RTI
        FUNCTN  ,R.W
        CLR     DATA
        CLR     BIT
        MOV     WADRA,R2
        MOV     WRDCNT,TTY
1S:    JSR     PC,GDATA
        CMP     DATA,(2)
        BEQ     .+4
        ERR3
        TST     (2)+
        TST     CFLAG
        BNE     2S
        INC     TTY
        BNE     1S
2S:    FUNCTN  ,CLEAR
        RTI

```

```

;CHECK DATA
;SKIP IF NO ERROR
;RETURN IF ERROR EXISTS
;REQUEST WINDOW IN R.W MODE
;SET UP DATA
;CLEAR FLAG
;SET UP BEGINNING OF BUFFER
;SET UP COUNT
;GET THE DATA
;SAME?
;SKIP IF OK
;DATA IS WRONG - TYPE IT
;ADD 2
;IS +C TYPED?
;BRANCH IF TYPED
;DEC THE COUNT
;LOOP IF NOT END
;REQUEST WINDOW IN CLEAR MODE
;RETURN

```

```

;THIS ROUTINE WILL GENERATE THE REQUESTED PATTERN.

```

```

GDATA: TST      MEMORY
        BMI     T3XOR9
        BGT     TBXOR13
T1XOR8: MOV     R2,R3
        SWAB   R3
        ROL   R3
        ADD   R2,R3
        ROR   R3
ALLOR:  ADD     BIT,R3
        ROR   R3
        BCS   1S
        COM   BIT
        COM   DATA
1S:    RTS     PC
TBXOR13:MOV    R2,R3
        MOV   R2,R1
        SWAB R3
        ROL  R1
        ROL  R1
        ROL  R1
        ROL  R1
T3X9:  ADD     R1,R3
        BR    ALLOR
T3XOR9:MOV    R2,R3
        ROR   R3
        MOV   R3,R1
        SWAB R1
        ROR   R3

```

```

;CHECK FLAG
;MM11G
;MM11F
;MM11E MEMORY
;MM11F MEMORY
;MM11G MEMORY

```


004

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
SUBROUTINES

MACY11 27(732) 05-OCT-76 11:05 PAGE 43

1452 007332 006003
1453 007334 000767

ROR R3
BR T3X9


```

1454                                     ;CHECK TO SEE IF HALT ON ERROR HAS BEEN SELECTED BY SWITCH 15.
1455                                     ;IF NOT CONTINUE TESTING.
1456
1457 007336 005037 177776      POTS:   CLR      @#PS      ;DROP PRIORITY
1458 007342 005737 177570      TST      @#SWR     ;HALT ON ERROR?
1459 007346 100001           BPL      .+4       ;NO - SKIP
1460 007350 000000           HALT
1461 007352 000002           RTI      ;WAIT FOR CONTINUE
1462                                     ;RETURN
1463
1464                                     ;ISSUE THE START COMMAND TO THE DEVICE UNDER TEST. THE WORDCOUNT
1465                                     ;REGISTER, THE BUS ADDRESS REGISTER, AND THE COMMAND REGISTER ARE
1466                                     ;LOADED. THE "MEX" BITS OF THE COMMAND REGISTER ARE ALSO LOADED
1467                                     ;AT THIS TIME FROM LOCATION "EMBITS".
1468 007354 017646 000000      .BEGIN: MOV     @ (6),-(6)      ;GET ADDRESS
1469 007360 017646 000000      MOV     @ (6),-(6)      ;GET ADDRESS
1470 007364 016776 002032 000000  MOV     IOWAD,@ (6)     ;LOAD MEMORY ADDRESS
1471 007372 022626           CMP     (6)+,(6)+      ;RESTORE STACK
1472 007374 062716 000002      ADD     #2,(6)         ;INCREMENT STACK
1473 007400 016705 001660      MOV     EMBITS,TTY     ;CLEAR JUNK
1474 007404 057605 000000      BIS     @ (6),TTY     ;GET REST OF COMMAND
1475 007410 062716 000002      ADD     #2,(6)         ;INCREMENT STACK
1476 007414 017646 000000      MOV     @ (6),-(6)     ;GET ADDRESS
1477 007420 017646 000000      MOV     @ (6),-(6)     ;GET ADDRESS
1478 007424 010576 000000      MOV     TTY,@ (6)     ;LOAD MEMORY ADDRESS
1479 007430 022626           CMP     (6)+,(6)+      ;RESTORE STACK
1480 007432 062716 000002      ADD     #2,(6)         ;INCREMENT STACK
1481 007436 000002           RTI
1482
1483 007440 000000      DATA: 0              ;DATA FOR MEMORY
1484 007442 000000      BIT: 0                ;FLAG
1485 007444 000000      MEMORY: 0            ;WHAT TYPE OF MEMORY
1486
1487
1488                                     ;REQUEST THE NEXT MEMORY SEGMENT FOR TESTING AND LOAD THE
1489                                     ;APPROPRIATE DATA PATTERN.
1490
1491 007446 104444           WRADR:  NEXT
1492 007450 104450 000031      FUNCTN  R.W          ;GET NEXT BANK
1493 007454 005067 177760      CLR     DATA        ;REQUEST WINDOW IN R.W MODE
1494 007460 005067 177756      CLR     BIT          ;CLEAR
1495 007464 016705 001734      MOV     WRDCNT,TTY   ;FLAGS
1496 007470 016702 001566      MOV     WADRA,R2     ;SET THE COUNTER
1497 007474 004767 177532      JSR     PC,GDATA    ;SET THE STARTING ADDRESS
1498 007500 016722 177734      MOV     DATA,@ (2)+ ;GET WORST CASE NOISE
1499 007504 005205           INC     TTY          ;PUT INTO MEMORY
1500 007506 001372           BNE     IS          ;DEC THE COUNTER
1501 007510 104450 000001      FUNCTN  ,CLEAR      ;NO - LOOP
1502 007514 000002           RTI      ;REQUEST WINDOW IN CLEAR MODE
1503                                     ;RETURN
1504
1505 172100      PARCSR= 172100
1506 000114      PARVEC= 114
1507 007516 012737 007610 000114 .MANK:  MOV     @#PARSR,@#PARVEC ;SET PARITY INTERRUPT VECTOR
1508 007524 012737 000340 000116      MOV     #340,@#PARVEC+2 ;SET PRIORITY LEVEL TO 7
1509 007532 013746 000004           MOV     @#4,-(SP)    ;SAVE CURRENT ERROR VECTOR

```



```

1510 007536 013746 000006          MOV      @#6,-(SP)          ;SAVE PRIORITY LEVEL
1511 007542 012737 000006 000004  MOV      @6,@#4
1512 007550 012737 000002 000006  MOV      @RTI,@#6
1513 007556 012700 172100          MOV      @PARCSR,R0      ;GET FIRST CSR ADDR
1514 007562 012702 000001          MOV      @1,R2
1515 007566 012720 000001          IS:     MOV      @1,(R0)+  ;SET ACTION ENABLE IF AVAILABLE
1516 007572 006302          ASL      R2              ;SHIFT AVAILABILITY INDICATOR
1517 007574 103374          BCC      IS
1518 007576 012637 000006          MOV      (SP)+,@#6      ;RESTORE ERROR VECTOR PRIORITY
1519 007602 012637 000004          MOV      (SP)+,@#4      ;AND INTERRUPT VECTOR
1520 007606 000207          RTS      PC
1521
1522 007610 000000          .PARSRV:HALT           ;MEMORY PARITY ERROR TRAP
1523 007612 000002          RTI
1524
1525          ;COME HERE IF A MEMORY MANAGEMENT VIOLATION OCCURS.
1526
1527          MMERR:
1528 007614 000004 007620          TYPE      +2
1529 007660 016705 167712          MOV      $R0+4,TTY
1530 007664 004767 002232          JSR      PC,PRINTR
1531 007670 000004 007674          TYPE      +2
1532 007710 016705 167656          MOV      $R0,TTY
1533 007714 004767 002202          JSR      PC,PRINTR
1534 007720 000000          HALT
1535 007722 000002          RTI
          ;.ASCIZ <15><12><12>"MEMORY MANAGEMENT ERROR AT "
          ;TYPE $R0+4 IN OCTAL
          ;TYPE LEADING ZERO'S
          ;.ASCIZ " $R0 = "
          ;TYPE $R0 IN OCTAL
          ;TYPE LEADING ZERO'S
          ;MM ERROR
          ;RETURN
    
```



```

1536                                     ;COMPLEMENT THE DATA BUFFER.
1537
1538 007724 016702 001332 WRCADR: MOV      WADRA,R2      ;SET UP BEGINNING OF BUFFER
1539 007730 104450 000031      ,R.W      ;REQUEST WINDOW IN R.W MODE
1540 007734 016705 001464      MOV      WRCNT,TTY  ;SET COUNTER
1541 007740 005122      15:  COM      (2)+    ;COMPLIMENT IT
1542 007742 005205      INC      TTY      ;COUNT IT
1543 007744 001375      BNE      15      ;LOOP TIL END OF CORE
1544 007746 104450 000001      FUNCTN ,CLEAR ;REQUEST WINDOW IN CLEAR MODE
1545 007752 000002      RTI                                     ;RETURN
1546
1547                                     ;REPORT A DATA COMPARE ERROR.
1548
1549 007754 032737 020000 177570 ER3: BIT      #SW13,#SWR  ;INHIBIT TYPEOUTS
1550 007762 001401      SEQ      .+4      ;SKIP IF TYPEOUTS WANTED
1551 007764 000002      RTI                                     ;RETURN
1552 007766 022767 041524 002056 CMP      #TC,INPUT ;IS IT A TC11
1553 007774 001003      BNE      .+10     ;SKIP IF NOT
1554 007776 012777 000001 001344 MOV      #1,#TCCS  ;STOP THE TC11
1555 010004 000004 010010      TYPE     .+2      ;ASCIZ <15><12>"DATA ERROR AT "
1556 010032 010205      MOV      R2,TTY   ;TYPE R2 WITH MX AS 18 BIT ADDRESS
1557 010034 004767 002116 JSR      PC,PRINTA ;GO TO ADDRESS PRINTER
1558 010040 000004 010044      TYPE     .+2      ;ASCIZ " TRUE = "
1559 010056 016705 177356      MOV      DATA,TTY ;TYPE DATA IN OCTAL
1560 010062 004767 002034 JSR      PC,PRINTR ;TYPE LEADING ZERO'S
1561 010066 000004 010072      TYPE     .+2      ;ASCIZ " RECEIVED = "
1562 010110 011205      MOV      (2),TTY  ;TYPE (2) IN OCTAL
1563 010112 004767 002004 JSR      PC,PRINTR ;TYPE LEADING ZERO'S
1564 010116 104434      STOP                                     ;HALT ON ERROR
1565 010120 000002      RTI                                     ;RETURN
1566
1567                                     ;THIS ROUTINE WILL TAKE THE CONTENTS OF LOCATION "MX" WHICH WAS
1568                                     ;SETUP BY SUBROUTINE "NXT" AND DETERMINE WHAT THE CONTENTS
1569                                     ;OF THE OTHER PROCESSOR'S RELOCATION REGISTER SHOULD BE. THIS INFORMATION
1570                                     ;IS PASSED TO THE OTHER PROCESSOR THROUGH THE WINDOW'S DATA BUFFER.
1571                                     ;IF DATA IS BEING TRANSFERRED TO SEGMENT 0 OF THE OTHER PROCESSOR
1572                                     ;THE STARTING ADDRESS OF THE WINDOW IS OFFSET BY 4K TO AVOID
1573                                     ;DESTROYING THE OTHER PROGRAM.
1574
1575 010122                                     SETBAK:
1576 010122 104450 000001      FUNCTN  CLEAR      ;REQUEST WINDOW IN CLEAR MODE
1577 010126 032767 004000 001132 BIT      #4000,SIZE ;4K WINDOW OR LESS?
1578 010134 001006      BNE      35      ;BRANCH IF YES
1579 010136 042767 020000 001116 BIC      #20000,WADRA
1580 010144 042767 020000 001250 BIC      #20000,IOWAD
1581 010152 016705 170636      35:  MOV      MX,TTY   ;GET THE MX
1582 010156 016700 001104      MOV      SIZE,R0  ;GET THE SIZE OF THE WINDOW
1583 010162 032767 004000 001076 BIT      #4000,SIZE ;4K OR LESS?
1584 010170 001402      BEQ      55      ;BRANCH IF GREATER
1585 010172 012700 174000      MOV      #174000,R0 ;MAKE WINDOW LOOK LIKE 4K
1586 010176 005400      55:  NEG      R0      ;MAKE IT A BIT
1587 010200 006000      15:  ROR      R0      ;SHIFT IT
1588 010202 103402      BCS      25      ;GET OUT IF SET
1589 010204 006305      ASL      TTY     ;MOVE INTO POSITION
1590 010206 000774      BR      15      ;LOOP
1591 010210 010577 001024      25:  MOV      TTY,#WDBRA ;PUT INTO DATA BUFFER

```


1592	010214	005767	170574		TST	MX		
1593	010220	001006			BNE	4S		
1594	010222	052767	020000	001032	BIS	#20000, WADRA		
1595	010230	052767	020000	001164	BIS	#20000, IOWAD		
1596	010236							
1597	010236	104450	000041	4S:	FUNCTN	, BANK		; REQUEST WINDOW IN BANK MODE
1598	010242	000002			RTI			; RETURN
1599								
1600								
1601	010244	012767	000000	162066	MAP:	MOV	#0, KIPAR0	; MAP BANK 0 INTO SELF
1602	010252	012767	077406	162020		MOV	#77406, KIPDR0	; 4K - R/W
1603	010260	012767	077406	162016		MOV	#77406, KIPDR2	; 4K - R/W
1604	010266	012767	077406	162012		MOV	#77406, KIPDR3	; 4K - R/W
1605	010274	012767	007600	162054		MOV	#7600, KIPAR7	; MAP BANK 7 INTO 37
1606	010302	012767	077406	162006		MOV	#77406, KIPDR7	; 4K - R/W
1607	010310	000002				RTI		; RETURN


```

1608 ;ALL OPERATIONS TO A DEVICE ARE UNDER INTERRUPT CONTROL AND WHEN A
1609 ;DEVICE IS BUSY PERFORMING AN OPERATION, CONTROL COMES HERE TO WAIT
1610 ;UNTIL AN INTERRUPT OCCURS.
1611
1612 010312 012737 010326 000004 WAIT: MOV    #TRAP4,2#4    ;SET TRAP VECTOR
1613 010320 005037 177776          CLR    2#PS          ;BR0
1614 010324 000777          BR     .             ;WAIT
1615
1616 ;COME HERE FOR A MEMORY TIMEOUT.
1617
1618 010326 005777 000704 TRAP4: TST    2#CSRA    ;CHECK FOR ERROR
1619 010332 100424          BMI    WINDOW      ;SKIP IF WINDOW ERROR
1620 010334 000004 010340      TYPE   +2          ;ASCIZ <15><12><12>"TRAP TO 4 FROM "
1621 010364 011600          MOV    (6),RO      ;GET RETURN ADDRESS
1622 010366 162700 000002      SUB    #2,RO       ;MAKE IT REAL ADDRESS
1623 010372 010005          MOV    RO,TTY      ;TYPE RO IN OCTAL
1624 010374 004767 001522      JSR   PC,PRINTR   ;TYPE LEADING ZERO'S
1625 010400 000000          HALT              ;TRAPPED TO 4
1626 010402 000002          RTI               ;RETURN
1627
1628 ;THIS IS THE WINDOW INTERRUPT HANDLER. IF THE INTERRUPT IS
1629 ;CAUSED BY THE SETTING OF NEW DATA, CONTROL IS TRANSFERRED TO "NOERR".
1630 ;IF IT IS AN ERROR, IT IS REPORTED TO THE OPERATOR.
1631
1632 010404 017700 000626 WINDOW: MOV    2#CSRA,RO ;SAVE WCSRA
1633 010410 100054          BPL    NOERR       ;SKIP IF NO ERROR FLAG
1634 010412 000004 010416      TYPE   +2          ;ASCIZ <15><12><12>"WINDOW ERROR AT "
1635 010442 011600          MOV    (6),RO      ;GET THE RETURN ADDRESS
1636 010444 162700 000002      SUB    #2,RO       ;MAKE IT REAL ADDRESS
1637 010450 010005          MOV    RO,TTY      ;TYPE RO IN OCTAL
1638 010452 004767 001444      JSR   PC,PRINTR   ;TYPE LEADING ZERO'S
1639 010456 000004 010462      TYPE   +2          ;ASCIZ " WCSRA = "
1640 010476 017705 000534      MOV    2#CSRA,TTY ;TYPE 2#CSRA IN OCTAL
1641 010502 004767 001414      JSR   PC,PRINTR   ;TYPE LEADING ZERO'S
1642 010506 000004 010512      TYPE   +2          ;ASCIZ " WDARA = "
1643 010526 017705 000512      MOV    2#DARA,TTY ;TYPE 2#DARA IN OCTAL
1644 010532 004767 001364      JSR   PC,PRINTR   ;TYPE LEADING ZERO'S
1645 010536 000000          HALT              ;WINDOW ERROR!
1646 010540 000002          RTI               ;RETURN
1647
1648 ;CONTROL COMES HERE WHEN THE OTHER PROCESSOR SETS NEW DATA. THE
1649 ;DATA BITS ARE EXAMINED TO DETERMINE WHICH FUNCTION IS BEING REQUESTED
1650 ;AND CONTROL IS TRANSFERRED TO THE PROPER ROUTINE.
1651
1652 010542 000300 NOERR: SWAB   RO      ;GET THE
1653 010544 042700 177761      BIC   8177761,RO  ;DATA BITS
1654 010550 000170 010554      JMP   2#WINTAB(0) ;GO TO ROUTINE
1655
1656
1657 010554 010576 WINTAB: W.NOP      ;NOP - CLOSE WINDOW
1658 010556 010626      W.READ   ;READ ONLY
1659 010560 010660      W.WRIT  ;WRITE ONLY
1660 010562 010660      W.R.W   ;READ/WRITE
1661 010564 010712      W.BANK  ;BANK SELECT
1662 010566 010574      W.FUTR  ;FUTURE USE
1663 010570 010574      W.FUTR  ;FUTURE USE

```


J04

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
WINDOW INTERRUPT HANDLER AND SCDULER

MACY11 27(732) 05-OCT--76 11:05 PAGE 49

1664 010572 010744

W.ZAP

;GET READY


```

1665 ;RESERVED FOR FUTURE USE.
1666
1667 010574 000000 W.FUTR: HALT ;RESERVED FOR FUTURE USE
1668
1669 ;WINDOW REQUESTED TO BE CLOSED.
1670
1671 010576 042777 000402 000432 W.NOP: BIC #402,2WCSRA ;CLOSE WINDOW
1672 010604 152777 000100 000424 BISR #100,2WCSRA ;I/E
1673 010612 005067 000456 CLR NOTRD ;NOT A READ
1674 010616 042777 017000 000412 BIC #17000,2WCSRA ;CLEAR NEW DAT
1675 010624 000002 RTI ;RETURN
1676
1677 ;WINDOW REQUESTED IN THE READ MODE.
1678
1679 010626 052777 000400 000402 W.READ: BIS #400,2WCSRA ;READ ONLY
1680 010634 152777 000100 000374 BISR #100,2WCSRA ;I/E
1681 010642 012767 177777 000424 MOV #-1,NOTRD ;SET READ FLAG
1682 010650 042777 017000 000360 BIC #17000,2WCSRA ;CLEAR NEW DATA
1683 010656 000002 RTI ;RETURN
1684
1685 ;WINDOW REQUESTED IN THE READ/WRITE OR WRITE MODE.
1686
1687 010660 W.R.W:
1688 010660 012767 177777 000406 W.WRIT: MOV #-1,NOTRD ;NOT A READ
1689 010666 052777 000402 000342 IS: BIS #402,2WCSRA ;READ/WRITE
1690 010674 152777 000100 000334 BISR #100,2WCSRA ;I/E
1691 010702 042777 017000 000326 BIC #17000,2WCSRA ;CLEAR NEW DATA
1692 010710 000002 RTI ;RETURN
1693
1694 ;LOAD THE CONTENTS OF THE DATA BUFFER INTO THE RELOCATION
1695 ;REGISTER.
1696
1697 010712 017777 000324 000326 W.BANK: MOV 2WDBRB,2WRARA ;BANK SELECT
1698 010720 042777 000402 000310 BIC #402,2WCSRA ;CLEAR NEW DATA
1699 010726 152777 000100 000302 BISR #100,2WCSRA ;I/E
1700 010734 042777 017000 000274 BIC #17000,2WCSRA ;CLEAR NEW DATA
1701 010742 000002 RTI ;RETURN
1702
1703 ;WARNING TO THIS PROCESSOR TO GET READY TO GET
1704 ;THE REQUESTED FUNCTION.
1705
1706 010744 042777 010502 000264 W.ZAP: BIC #10502,2WCSRA
1707 010752 032777 010000 000256 IS: BIT #10000,2WCSRA ;WAIT FOR NEW DATA
1708 010760 001774 BEQ IS ;TO GO AWAY
1709 010762 000167 177416 JMP WINDOW ;NOW GET FUNCTION
1710
1711
1712 ;DETERMINE WHERE THE WINDOW'S CSR REGISTER IS LOCATED.
1713
1714 010766 012700 164000 REGSCN: MOV #164000,R0 ;SET WCSRA FOR SCANNING
1715 010772 012702 000010 MOV #0,R2 ;SET COUNT
1716 010776 012737 011010 000004 MOV #55,2#4 ;SET FOR TIMEOUT
1717 011004 005710 6S: TST (0) ;IS IT THERE?
1718 011006 000207 RTS PC ;RETURN
1719 011010 022626 5S: CMP (6)+,(6)+ ;CLEAR STACK
1720 011012 062700 000020 ADD #20,R0 ;BUMP ADDRESS

```



```

1721 011016 005302
1722 011020 001371
1723 011022 000000
1724 011024 000760
1725
1726
1727
1728
1729
1730
1731 011026 000005
1732 011030 012706 000500
1733 011034 012767 177777 000230
1734 011042 012767 000777 000014
1735 011050 004767 177712
1736 011054 010067 000040
1737 011060 012710 000402
1738 011064 000777
1739 011066 005077 000026
1740 011072 012737 000240 006660
1741 011100 012737 000240 006662
1742 011106 012737 000240 007106
1743 011114 000137 001026
1744
1745 011120 000000

```

```

DEC R2 ;DEC COUNT
BNE 65 ;LOOP UNTIL FOUND
HALT ;NO WCSRA
BR REGSCN ;TRY IT AGAIN

```

```

;OPEN THE WINDOW TO ALLOW THE OTHER PROCESSOR TO LOAD THE PROGRAM
;INTO ITSELF. THIS PROCESSOR WILL LOOP AT LOCATION "BR." WHILE THE
;OTHER PROCESSOR GETS THE PROGRAM. THE OTHER PROCESSOR WILL REPLACE THE
;THE BRANCH TO SELF INSTRUCTION WITH A NOP WHEN THE PROGRAM IS LOADED.

```

```

OPEN: RESET ;ZAP THE WORLD
MOV #500, SP ;INITIALIZE THE STACK
MOV #-1, ONCE ;RESET ONCE
MOV #777, BR. ;RESTORE THE BR.
JSR PC, REGSCN ;GET THE REGISTER
MOV RO, BR.1 ;SAVE WCSRA
MOV #402, (0) ;OPEN THE WINDOW
BR. ;WAIT
CLR @BR.1 ;CLEAR WCSRA
MOV #240, @DOF2
MOV #240, @DOF2+2
MOV #240, @DOF5
JMP @BEGIN ;SELF START

BR.1: 0

```


1746	011122	116767	166434	000722	READR:	MOV	177562, INPUT	; READ THE CHARACTER
1747	011130	142767	000200	000714		BIC	#200, INPUT	; CLEAR JUNK
1748	011136	122767	000003	000706	IS:	CMP	#'C-100, INPUT	; IS IT A 'C'?
1749	011144	001401				BEO	.+4	; SKIP IF 'C'
1750	011146	000002				RTI		; RETURN IF NOT
1751	011150	010667	000010			MOV	SP, CFLAG	; SET CONTROL "C" FLAG
1752	011154	042737	000100	177560		BIC	#100, @#177560	; DISABLE TTY INTERRUPT
1753	011162	000002				RTI		
1754								
1755	011164	000000			CFLAG:		0	
1756								
1757								
1758	011166	000000			CHK:		0	
1759	011170	000000			EX:		0	
1760	011172	020040	051503	036440	M3:	.ASCIZ	" CS = "	
1761	011200	000040						
1762	011202	020040	051105	036440	M4:	.ASCIZ	" ER = "	
1763	011210	000040						
1764						.EVEN		
1765	011212	032456	030440	031040	WIND:	.ASCII	" .5 1 2 4 8 16 32 "	
1766	011220	032040	034040	033061				
1767	011226	031063						
1768	011230	000000	000000		SIZ:		0,0	
1769	011234	000000			OFFSET:		0	
1770								
1771		177572			SRO=		177572	;KT11 - MEMORY MANAGEMENT
1772		172340			KIPAR0=		172340	
1773		172300			KIPDR0=		172300	
1774		172344			KIPAR2=		172344	
1775		172304			KIPDR2=		172304	
1776		172346			KIPAR3=		172346	
1777		172306			KIPDR3=		172306	
1778		172356			KIPAR7=		172356	
1779		172316			KIPDR7=		172316	
1780								
1781	011236	164000			WCSRA:		164000	; CONTROL & STATUS REG. (R/W) - WINDOW A
1782	011240	164002			WDBRA:		164002	; OUTPUT DATA BUFFER (R/W)
1783	011242	164004			WDBRB:		164004	; INPUT DATA BUFFER (R)
1784	011244	164006			WDARA:		164006	; DISPLACEMENT ADDRESS REG. (R)
1785	011246	164010			WRARA:		164010	; RELOCATION ADDRESS REGISTER (R/W)
1786	011250	164012			WADRAX:		164012	; WINDOW ADDRESS REGISTER (R)
1787	011252	164014			WVARA:		164014	; VECTOR ADDRESS REGISTER (R)
1788	011254	000000					0	
1789								
1790	011256	000300	000302		WVECA:		300,302	; VECTOR ADDRESS
1791								
1792	011262	020000			WADRA:		20000	; WINDOW ADDRESS
1793	011264	000000			EMBITS:		0	; EXTENDED MEMORY BITS
1794								
1795	011266	000000			SIZE:		0	; WINDOW SIZE
1796	011270	000000			MMN:		0	; MEMORY MANAGEMENT REQUIRED FLAG
1797	011272	177777			ONCE:		-1	; ONCE ONLY FLAG
1798	011274	000000			NOTRD:		0	; READ FLAG - 0=NO READ

1799	011276	177460		RFCS:	177460		
1800	011300	177470		RFER:	177470		;RF11 DISK (256K)
1801	011302	177462		RFC:	177462		
1802	011304	177464		RFCA:	177464		
1803	011306	177466		RFDA:	177466		
1804	011310	177446		RCCS:	177446		;RC11 DISK (64K)
1805	011312	177444		RCER:	177444		
1806	011314	177450		RCWC:	177450		
1807	011316	177452		RCCA:	177452		
1808	011320	177442		RCD:	177442		
1809	011322	176714		RPCS:	176714		;RP11 DISK (PACK)
1810	011324	176716		RPWC:	176716		
1811	011326	176720		RPCA:	176720		
1812	011330	176724		RPDA:	176724		
1813	011332	176712		RPER:	176712		
1814	011334	176710		RPOS:	176710		
1815	011336	177404		RKCS:	177404		;RK11 DISK (CARTRIDGE)
1816	011340	177402		RKER:	177402		
1817	011342	177406		RKWC:	177406		
1818	011344	177410		RKCA:	177410		
1819	011346	177412		RKDA:	177412		
1820	011350	177342		TCCS:	177342		;TC11 DECTAPE
1821	011352	177340		TCER:	177340		
1822	011354	177344		TCWC:	177344		
1823	011356	177346		TCCA:	177346		
1824	011360	177350		TCDB:	177350		
1825	011362	172522		TMCS:	172522		;TM11 MAGTAPE
1826	011364	172520		TMER:	172520		
1827	011366	172524		TMBC:	172524		
1828	011370	172526		TMCA:	172526		
1829							
1830	011372	000204	000206	RFTRAP:	204, 206		
1831	011376	000210	000212	RCTRAP:	210, 212		
1832	011402	000254	000256	RPTRAP:	254, 256		
1833	011406	000220	000222	RKTRAP:	220, 222		
1834	011412	000214	000216	TCTRAP:	214, 216		
1835	011416	000224	000226	TMTRAP:	224, 226		
1836							
1837	011422	000000		IOWAD:	0		
1838	011424	177400		WROCNT:	-400		;WORD COUNT


```

1839
1840
1841
1842
1843
1844
1845
1846
1847
1848 011426 010001
1849 011430 022701 000006
1850 011434 100002
1851 011436 012701 000006
1852 011442 006001
1853 011444 006001
1854 011446 006001
1855 011450 006001
1856 011452 042701 017777
1857 011456 012702 017200
1858 011462 060201
1859 011464 010167 000034
1860 011470 012122
1861 011472 022702 020000
1862 011476 001374
1863 011500 000207
1864
1865 011502 012702 017200
1866 011506 016701 000012
1867 011512 012221
1868 011514 022702 020000
1869 011520 001374
1870 011522 000207
1871
1872 011524 000000
1873
1874
1875
1876
1877
1878 011526 005767 177432
1879 011532 001001
1880 011534 000207
1881 011536 012706 000500
1882 011542 000004 011546
1883 011552 005067 177406
1884 011556 104450 000001
1885 011562 005037 177776
1886 011566 000137 002564

```

```

; SLOADR GET AND RESTORE THE LOADER
; THESE ROUTINES FIRST FIND THE LOADER (TOP OF HIGHEST
; 4K BANK IN 28K) AND SAVE IT AT TAG "ENDP:". ENTRY FOR
; THIS ROUTINE IS "JSR PC,LODGET".
; "JSR PC,LODRES" WILL RESTORE THE LOADER IN ITS ORIGINAL
; LOCATION. ".LOD:" CONTAINS THE ADDRESS OF THE LOADER.

LODGET: MOV R0,R1 ;GET THE BANK
        CMP #6,R1 ;IS IT > 6?
        BPL IS ;SKIP IF LES THAN 6
        MOV #6,R1 ;IF > 7 MAKE 7
1S: ROR R1 ;GET THE
    ROR R1 ;UPPER
    ROR R1 ;THREE
    ROR R1 ;BITS
    BIC #17777,R1 ;CLEAR JUNK
    MOV #ENDP,R2 ;GET SAVE ADDRESS
    ADD R2,R1 ;MAKE OTHER ONE
    MOV R1,.LOD ;SAVE ADDRESS
2S: MOV (1)+(2)+ ;MOVE WORD
    CMP #20000,R2 ;END?
    BNE 2S ;NO!
    RTS PC ;RETURN

LODRES: MOV #ENDP,R2 ;GET END OF PROGRAM
        MOV .LOD,R1 ;GET SAVE ADDRESS
1S: MOV (2)+(1)+ ;RESTORE WORD
    CMP #20000,R2 ;END?
    BNE 1S ;LOOP
3S: RTS PC ;RETURN

.LOD: 0 ;STARTING ADDRESS OF LOADER

; THIS ROUTINE TESTS THE FLAG TO SEE IF CONTROL "C" HAS BEEN TYPED.
; IF TYPED THE PROGRAM RESTARTS, OTHERWISE IT CONTINUES.

CONTC: TST CFLAG ;CONTROL "C" TYPED?
        BNE 1S ;BRANCH IF YES
        RTS PC ;CONTINUE
1S: MOV #500,SP ;RESTORE THE STACK
    TYPE A+2 ;ASCIZ "C"(177)
    CLR CFLAG ;CLEAR RESTART FLAG
    FUNCTN CLEAR ;REQUEST WINDOW IN CLEAR MODE
    CLR #PS
    JMP #REDO ;GO RESTART

```



```

1887
1888
1889
1890
1891
1892
1893
1894
1895 011572 010467 000146
1896 011576 010546
1897 011600 017605 000002
1898 011604 032705 177400
1899 011610 001004
1900 011612 010567 000124
1901 011616 012705 011742
1902 011622 105715
1903 011624 001423
1904 011626 122715 000012
1905 011632 001012
1906 011634 116704 000101
1907 011640 116737 000074 177566
1908 011646 105737 177564
1909 011652 100375
1910 011654 005304
1911 011656 001370
1912 011660 112537 177566
1913 011664 105737 177564
1914 011670 100375
1915 011672 000753
1916 011674 017646 000002
1917 011700 062766 000002 000004
1918 011706 022666 000002
1919 011712 001006
1920 011714 062705 000002
1921 011720 042705 000001
1922 011724 010566 000002
1923 011730 012605
1924 011732 016704 000006
1925 011736 000002
1926 011740 001000
1927 011742 000000
1928 011744 000000
1929
1930 011746 010346
1931 011750 012703 012052
1932 011754 022703 012072
1933 011760 001412
1934 011762 105737 177560
1935 011766 100375
1936 011770 113713 177562
1937 011774 142713 000200
1938 012000 122713 000177
1939 012004 001005
1940 012006
1941 012006 000004 012012
1942 012016 000754

```

; STYPE MESSAGE TIMEOUT ROUTINE
; THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE
; CALL CAN BE IN ONE OF 3 FORMS: 1) "TYPE ADR" - TYPES THE
; MESSAGE STARTING IN LOCATION "ADR:" 2) "TYPE CHAR" - TYPES
; THE ASCII "CHAR", AND 3) "PRINT <<15><12>"MESSAGE"> - TYPES
; THE MESSAGE WHICH IS INLINE ASCII.

```

IOTS:  MOV R4,IOTS          ;SAVE R4
        MOV TTY,-(6)       ;SAVE TTY R5=TTY
        MOV 32(6),TTY      ;GET ADDRESS TO BE TYPED
        BIT 3177400,TTY    ;IS IT A TYPEN?
        BNE 15             ;NO
        MOV TTY,TYPE      ;GET THE CHARACTER
        MOV 8,TYPE,TTY    ;FUDGE THE ADDRESS
15:     TSTB (TTY)         ;TERMINATOR?
        BEQ 25            ;GET OUT IF SO
        CMPB 312,(TTY)    ;IS THE CHAR A LINE FEED?
        BNE 45            ;BRANCH IF NO
        MOVB FILCHR+1,R4  ;GET FILLER COUNT
55:     MOVB FILCHR,3177566 ;OUTPUT FILLER CHAR
        TSTB 3177564      ;WAIT FOR DONE
        BPL -4
        DEC R4             ;DECREMENT FILLER COUNT
        BNE 55
45:     MOVB (TTY)+,3177566 ;LOAD AND TYPE THE CHARACTER
        TSTB 3177564      ;IS THE PRINTER READY
        BPL -4           ;WAIT UNTIL IT IS
        BR 15             ;GET THE NEXT CHARACTER
25:     MOV 32(6),-(6)     ;GET ADDRESS TO BE TYPED
        ADD 32,4(6)       ;ADD 2 TO THE ADDRESS
        CMP (6)+,2(6)    ;IS IT .+2?
        BNE 35            ;NO
        ADD 32,TTY        ;ADD 2 TO THE ADDRESS
        BIC 31,TTY        ;BACK UP TO AN EVEN BYTE
        MOV TTY,2(6)     ;RESTORE ADDRESS
35:     MOV (6)+,TTY      ;RESTORE TTY
        MOV IOTS,R4      ;RESTORE R4
        RTI              ;RETURN
        FILCHR: 1000     ;FILCHR=0 (CHAR) FILCHR+1=2 (COUNT)
        TYPE: 0          ;CHARACTER TYPE LOCATION
        IOTS: 0
READS:  MOV R3,-(6)       ;SAVE R3
15:     MOV 3INPUT,R3     ;GET ADDRESS
        CMP 3INPUT+20,R3 ;BUFFER FULL?
        BEQ 45            ;YES - TYPE "?"
25:     TSTB 3177560      ;WAIT FOR
        BPL -4           ;A CHARACTER
        MOVB 3177562,(3) ;GET CHARACTER
        BICB 3200,(3)    ;GET RID OF JUNK
        CMPB 3177,(3)   ;IS IT A RUBOUT
        BNE 35            ;SKIP IF NOT
45:     TYPE 15+2        ;ASCIZ "?"<15><12>
        BR 15            ;ZAP THE BUFFER AND LOOP

```


MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER
TTY INPUT ROUTINE

MACY11 27(732) 05-OCT-76 11:05 PAGE 56

1943	012020	111367	177716
1944	012024	000004	011742
1945	012030	122723	000015
1946	012034	001352	
1947	012036	105063	177777
1948	012042	000004	000012
1949	012046	012603	
1950	012050	000207	
1951	012052	000020	

```

3S:  MOVB  (3),TYPE      ;SET UP FOR TYPING
      TYPE  TYPE        ;ECHO IT
      CMPB  15,(3)+     ;CHECK FOR RETURN
      BNE   25          ;LOOP IF NOT RETURN
      CLRB  -1(3)       ;ZAP RETURN (THE 15)
      TYPE  12         ;TYPE A LINE FEED
      MOV   (6)+,R3     ;RESTORE R3
      RTS   PC          ;RETURN
      PC    20          ;TTY INPUT AREA

INPUT:.BLKW

```



```

1952 ;          SOCTAL          OCTAL TYPEOUT ROUTINE
1953
1954 ; THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
1955 ; ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, TYPE AN 18 BIT ADDRESS, OR TYPE
1956 ; THE 16 BITS. IT IS CALLED VIA THE DUMP, S0UMP, DUMP18, OR BITYPE MACRO'S.
1957
1958 012112 012767 170101 000202 BITYPS: MOV      #170101,.PR      ;SET BIT FLAG ANS 16. CHARACTER COUNT
1959 012120 000411                BR          .PTIT        ;NOW TYPE IT IN BIT FORM
1960 012122 112767 000001 000172 PRINTR: MOVB   #1,.PR          ;SET ZERO FILL SWITCH
1961 012130 000402                BR          .+6          ;SKIP
1962 012132 005067 000164                PRINTS: CLR      .PR          ;SUPPRESS LEADING ZERO'S
1963 012136 112767 177772 000157          MOVB   #-6,.PR+1      ;SET COUNT
1964 012144 010446                .PTIT:  MOV      R4,-(6)      ;SAVE R4
1965 012146 012704 012324          MOV      #.PR+2,R4    ;SET POINTER TO FIRST ASCII CHAR.
1966 012152 105014                CLRB      (4)         ;CLEAR FIRST BYTE
1967 012154 000432                BR          .PRF        ;ROTATE FIRST BIT
1968 012156 010446                PRINTA: MOV     R4,-(6)  ;SAVE R4
1969 012160 012704 012324          MOV      #.PR+2,R4    ;SET UP POINTER TO OUTPUT AREA
1970 012164 116714 166624          MOVB   MX,(4)         ;MX CONTAINS UPPER 5 BITS
1971 012170 006305                ASL      TTY          ;GET RID
1972 012172 006305                ASL      TTY          ;OF 3
1973 012174 006305                ASL      TTY          ;JUNK BITS
1974 012176 106214                ASRB     (4)         ;GET BIT13
1975 012200 006005                ROR      TTY          ;PACK IT
1976 012202 106214                ASRB     (4)         ;GET BIT14
1977 012204 006005                ROR      TTY          ;PACK IT
1978 012206 152724 000060          BISB     #'0,(4)+     ;MAKE IT ASCII
1979 012212 012767 175401 000102          MOV      #175401,.PR  ;-5, 1 - 5 BYTES AND FILL
1980 012220 105014                .PRL:  CLRB      (4)         ;CLEAR BYTE OF CHARACTER
1981 012222 032767 000100 000072          BIT      #100,.PR     ;BIT TYPING MODE?
1982 012230 001004                BNE      .PRF        ;YES - SKIP 2 ROTATES
1983 012232 006105                ROL      TTY          ;ROTATE BIT INTO C
1984 012234 106114                ROLB     (4)         ;PACK IT
1985 012236 006105                ROL      TTY          ;ROTATE BIT INTO C
1986 012240 106114                ROLB     (4)         ;PACK IT
1987 012242 006105                .PRF:  ROL      TTY          ;ROTATE BIT INTO C
1988 012244 106114                ROLB     (4)         ;PACK IT
1989 012246 105714                TSTB     (4)         ;IS IT ZERO?
1990 012250 001402                BEQ      .+6         ;SKIP INC
1991 012252 105267 000044          INCB     .PR          ;SET FILL SWITCH
1992 012256 105767 000040          TSTB     .PR          ;CHECK FILL SWITCH
1993 012262 001402                BEQ      .+6         ;SKIP BITSET
1994 012264 152724 000060          BISB     #'0,(4)+     ;MAKE INTO ASCII CHAR
1995 012270 105267 000027          INCB     .PR+1        ;INC COUNT
1996 012274 001351                BNE      .PRL        ;REPEAT
1997 012276 022704 012324          CMP      #.PR+2,R4    ;EMPTY BUFFER?
1998 012302 001002                BNE      .+6         ;SKIP IF NOT
1999 012304 112724 000060          MOVB   #'0,(4)+     ;LOAD 1 ZERO
2000 012310 105014                CLRB     (4)         ;NULL TERMINATOR
2001 012312 000004 012324          TYPE     .PR+2        ;TYPE IT
2002 012316 012604                MOV      (6)+,R4      ;RESTORE R4
2003 012320 000207                RTS      PC           ;RETURN
2004 012322 000012                .PR:    .BLKW     12   ;COUNT, SWITCH, AND OUTPUT BUFFER

```


F05

MAINDEC-11-DZDAB-A
DZDABA.P11

BUS WINDOW EXERCISER MACY11 27(732) 05-OCT-76 11:05 PAGE 58
POWER DOWN AND UP ROUTINES

2005	012346	012777	012474	000126	PDOWN:	MOV	#ILLUP, 2PUVECS	:SET FOR FAST UP
2006	012354	012777	000340	000122		MOV	#340, 2PUVECS+2	:PRIO:7
2007	012362	010046				MOV	R0, -(6)	:PUSH R0 ON STACK
2008	012364	010146				MOV	R1, -(6)	:PUSH R1 ON STACK
2009	012366	010246				MOV	R2, -(6)	:PUSH R2 ON STACK
2010	012370	010346				MOV	R3, -(6)	:PUSH R3 ON STACK
2011	012372	010446				MOV	R4, -(6)	:PUSH R4 ON STACK
2012	012374	010546				MOV	R5, -(6)	:PUSH R5 ON STACK
2013	012376	010667	000076			MOV	SP, SAVR6	:SAVE SP
2014	012402	012777	012412	000072		MOV	#PUPS, 2PUVECS	:SET UP VECTOR
2015	012410	000000				HALT		:WAIT FOR PF
2016								
2017	012412	016706	000062		PUPS:	MOV	SAVR6, SP	:GET SP
2018	012416	005001				CLR	R1	:WAIT LOOP FOR THE TTY
2019	012420	005201			IS:	INC	R1	:WAIT FOR THE INC
2020	012422	001376				BNE	IS	:OF WORD
2021	012424	012605				MOV	(6)+, R5	:POP STACK INTO R5
2022	012426	012604				MOV	(6)+, R4	:POP STACK INTO R4
2023	012430	012603				MOV	(6)+, R3	:POP STACK INTO R3
2024	012432	012602				MOV	(6)+, R2	:POP STACK INTO R2
2025	012434	012601				MOV	(6)+, R1	:POP STACK INTO R1
2026	012436	012600				MOV	(6)+, R0	:POP STACK INTO R0
2027	012440	012737	012346	000024		MOV	#PDOWN, 2#24	:SET UP THE POWER DOWN VECTOR
2028	012446	012737	000340	000026		MOV	#340, 2#26	:PRIO:7
2029	012454	000004	012460			TYPE	+2	:ASCIZ <15><12>"POWER"
2030	012470	000167	170360			JMP	RTPF	:JMP TO USER ADDRESS
2031								
2032	012474	000000			ILLUP:	HALT		:THE POWER UP SEQUENCE WAS STARTED
2033	012476	011062				BR.-2		:BEFORE THE POWER DOWN WAS COMPLETE
2034								
2035	012500	000000			SAVR6:	0		:PUT THE SP HERE
2036	012502	000024	000026		PUVECS:	24, 26		:POWER UP VECTOR
2037	012506	000010			PATCH1:	.BLKW 10		:PATCH AREA #1
2038	012526	000010			PATCH2:	.BLKW 10		:PATCH AREA #2
2039								
2040		017200			.	= 17200		
2041								
2042	017200	000300			ENDP:	.BLKB 300		
2043								
2044		000001			.	END		

ADD	506	522	555	595	644	789	1089	1139	1256	1385	1430	1432	1445	1472	1475
	1480	1720	1858	1917	1920										
ASL	526	604	609	610	616	619	1196	1220	1371	1516	1589	1971	1972	1973	
ASR	590	594	707	711	712	714	715	716	1361	1362					
ASRB	1974	1976													
BCC	617	620	1517												
BCS	1434	1588													
BEQ	435	486	541	593	665	669	673	682	706	710	1300	1304	1310	1320	1324
	1344	1346	1352	1366	1381	1383	1401	1410	1550	1584	1708	1749	1903	1933	1990
	1993														
BGT	524	1091	1158	1425											
BIC	528	640	678	686	687	723	1325	1364	1579	1580	1653	1671	1674	1682	1691
	1698	1700	1706	1752	1856	1921									
BICB	1355	1747	1937												
BIS	618	621	679	722	724	728	733	738	743	748	754	1474	1594	1595	1679
	1689														
BISB	1350	1356	1672	1680	1690	1699	1978	1994							
BIT	434	488	664	668	672	684	703	705	709	870	932	1000	1065	1122	1235
	1291	1295	1299	1303	1319	1323	1345	1347	1351	1357	1382	1549	1577	1583	1707
	1898	1981													
BLT	653	655	1294												
BMI	1151	1168	1258	1309	1424	1619									
BNE	433	489	496	510	557	583	607	632	647	685	704	726	731	736	741
	746	751	757	787	871	933	1001	1066	1123	1154	1171	1175	1236	1292	1296
	1348	1358	1414	1416	1500	1543	1553	1578	1593	1722	1862	1869	1879	1899	1905
	1911	1919	1939	1946	1982	1996	1998	2020							
BPL	868	930	988	990	998	1012	1063	1077	1120	1134	1149	1166	1233	1459	1633
	1850	1909	1914	1935											
BR	511	547	553	560	574	596	657	667	671	708	713	794	992	1155	1159
	1172	1261	1312	1314	1316	1354	1384	1446	1453	1590	1614	1724	1738	1915	1942
	1959	1961	1967												
CLR	430	461	462	480	482	483	484	494	505	529	591	602	614	626	676
	698	721	762	769	792	827	832	844	857	865	889	894	906	919	927
	948	953	965	978	991	995	1022	1027	1039	1052	1060	1117	1190	1230	1311
	1342	1353	1404	1405	1457	1493	1494	1613	1673	1739	1883	1885	1962	2018	
CLRB	1947	1966	1980	2000											
CMP	432	485	509	523	548	554	576	582	652	654	725	730	735	740	745
	750	756	786	1090	1157	1174	1257	1293	1365	1409	1471	1479	1552	1719	1849
	1861	1868	1918	1932	1997										
CMPB	1748	1904	1938	1945											
COM	1435	1436	1541												
COMB	492	1313													
DEC	531	556	606	631	646	1721	1910								
EMT	384														
HALT	402	436	446	559	575	656	1260	1379	1460	1522	1534	1625	1645	1667	1723
	2015	2032													
INC	507	518	534	869	931	999	1064	1121	1234	1301	1317	1415	1499	1542	2019
INCB	1991	1995													
IOT	385														
JMP	405	413	417	418	419	438	487	729	734	739	744	749	755	759	763
	773	1176	1210	1259	1654	1709	1743	1886	2030						
JSR	445	478	535	538	572	613	662	694	695	790	793	828	875	878	890
	937	940	949	1005	1008	1023	1070	1073	1088	1127	1130	1191	1240	1243	1307
	1326	1375	1378	1408	1497	1530	1533	1557	1560	1563	1624	1638	1641	1644	1735
MOV	410	411	412	427	431	437	463	464	465	466	467	468	469	470	471
	472	473	474	475	476	477	479	481	490	493	503	504	519	520	525

	532	533	537	543	545	549	550	551	571	577	578	579	581	584	585
	586	587	597	598	601	603	608	611	612	615	622	623	625	627	628
	629	630	633	634	635	641	642	643	645	648	649	650	658	660	661
	663	666	670	674	675	677	680	683	692	696	701	702	719	720	727
	732	737	742	747	752	753	758	780	783	784	785	788	830	831	842
	843	855	856	872	877	892	893	904	905	917	918	934	939	951	952
	963	964	976	977	986	1002	1007	1010	1025	1026	1037	1038	1050	1051	1067
	1072	1075	1084	1092	1096	1097	1109	1110	1124	1129	1132	1140	1141	1143	1145
	1161	1162	1194	1195	1204	1205	1207	1218	1219	1237	1242	1253	1298	1302	1306
	1318	1322	1349	1359	1360	1363	1367	1368	1369	1374	1377	1406	1407	1427	1438
	1439	1447	1449	1468	1469	1470	1473	1476	1477	1478	1495	1496	1498	1507	1508
	1509	1510	1511	1512	1513	1514	1515	1518	1519	1529	1532	1538	1540	1554	1556
	1559	1562	1581	1582	1585	1591	1601	1602	1603	1604	1605	1606	1612	1621	1623
	1632	1635	1637	1640	1643	1681	1688	1697	1714	1715	1716	1732	1733	1734	1736
	1737	1740	1741	1742	1751	1848	1851	1857	1859	1860	1865	1866	1867	1881	1895
	1896	1897	1900	1901	1916	1922	1923	1924	1930	1931	1949	1958	1964	1965	1966
	1969	1979	2002	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2017	2021
	2022	2023	2024	2025	2026	2027	2028								
	1255	1746	1906	1907	1912	1936	1943	1960	1963	1970	1999				
MOV8	588	1586													
NEG	798	800	803	805	806	808	811	813							
NEGB	437														
NOP	428	1731													
RESET	605	1373	1429	1441	1442	1443	1444	1983	1985	1987					
ROL	801	802	809	810	1984	1986	1988								
ROLB	801	802	809	810	1984	1986	1988								
ROR	636	637	638	639	1431	1433	1448	1451	1452	1587	1852	1853	1854	1855	1975
	1977														
RORB	799	804	807	812											
RTI	636	848	861	880	898	910	923	942	957	969	982	1013	1031	1043	1056
	1078	1100	1113	1135	1146	1163	1200	1208	1224	1245	1328	1386	1402	1419	1461
	1481	1502	1512	1523	1535	1545	1551	1565	1598	1607	1626	1646	1675	1683	1692
	1701	1750	1753	1925											
RTS	814	1437	1520	1718	1863	1870	1880	1950	2003						
SEC	1372														
SUB	429	1142	1254	1622	1636										
SWAB	527	589	1315	1428	1440	1450	1652								
TRAP	383	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277
	1278	1279	1280	1281	1282	1283	1284								
TST	491	495	508	521	540	546	552	580	592	624	659	681	867	929	989
	997	1062	1119	1144	1148	1150	1153	1165	1167	1170	1206	1232	1308	1343	1380
	1400	1412	1413	1423	1458	1592	1618	1717	1878						
TSTB	987	1011	1076	1133	1902	1908	1913	1934	1989	1992					
.ASCII	1765														
.ASCIZ	537	540	543	545	559	571	574	601	652	694	762	771	773	874	936
	1004	1069	1126	1239	1306	1322	1371	1377	1529	1532	1556	1559	1562	1621	1635
	1640	1643	1760	1762	1883	1942	2030								
.BLKB	2042														
.BLKW	1951	2004	2037	2038											
.ENABL	1	323													
.END	2044														
.ENDC	399	873	880	935	942	1003	1013	1068	1078	1125	1135	1238	1245	1849	1866
	1870	1902	1926	1933	1952	1980	2005	2013	2021	2030	2031				
.EVEN	537	540	543	545	559	571	574	601	652	694	762	771	773	874	936
	1004	1069	1126	1239	1306	1322	1371	1377	1529	1532	1556	1559	1562	1621	1635
	1640	1643	1764	1883	1942	2030									
.IF	399	873	880	935	942	1003	1010	1068	1075	1125	1132	1238	1245	1848	1865

	1870	1898	1916	1932	1951	1952	1968	2005	2013	2021	2029	2030				
.IFF	1848	1865	1926	1933	1952	2030										
.IIF	331	332	333	334	335	336	337	338	339	1951	2005					
.IRP	2007	2021														
.LIST	1	323	399	402	447	537	540	543	545	559	571	574	601	652	688	
	694	762	771	773	774	819	865	874	881	927	936	943	995	1004	1014	
	1060	1069	1079	1117	1126	1182	1230	1239	1246	1264	1265	1266	1267	1268	1269	
	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279	1280	1281	1282	1283	1284	
	1285	1286	1306	1322	1371	1377	1529	1532	1556	1559	1562	1608	1621	1635	1640	
	1643	1799	1839	1883	1887	1930	1942	1952	2005	2030						
.MACRO	1	447	1246													
.MCALL	323	399														
.NLIST	1	323	399	402	447	537	540	543	545	559	571	574	601	652	688	
	694	762	771	773	774	819	865	874	881	927	936	943	995	1004	1014	
	1060	1069	1079	1117	1126	1182	1230	1239	1246	1264	1265	1266	1267	1268	1269	
	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279	1280	1281	1282	1283	1284	
	1285	1286	1306	1322	1371	1377	1529	1532	1556	1559	1562	1608	1621	1635	1640	
	1643	1799	1839	1883	1887	1930	1942	1952	2005	2030						
.PAGE	993	1246	1887	1952	2005											
.REM	1															
.REPT	402															
.SBTTL	447	688	774	819	865	881	927	943	995	1014	1060	1079	1117	1182	1230	
	1246	1286	1608	1799	1839	1887	1930	1952	2005							
.TITLE	323															

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*.DZDABA.SEQ/SOL/CRF/PAGNUM/NL:TOC=STEVE.SML,DZDABA.P11
 RUN-TIME: 22 28 3 SECONDS
 RUN-TIME RATIO: 369/54=6.7
 CORE USED: 26K (51 PAGES)

