

KDJ11-B

EAROM MAINT UTILITY
COKDBAO

COPYRIGHT (c) 1984
AH-T869A-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA

.REM E

IDENTIFICATION

PRODUCT CODE: AC-T868A-MC
 PRODUCT NAME: COKDBAO EAROM MAINT. UTILITY
 PRODUCT DATE: APRIL 1984
 MAINTAINER: LOW END DIAGNOSTICS ENGINEERING
 AUTHOR: RUSSELL YOUNG

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

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1984 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

E

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45

```
47 ;THIS PROGRAM IS BASED ON ONE WRITTEN BY NESTOR HERNANDEZ.
48
49 ;NOTE: THE FILE I/O ALL WORKS CORRECTLY, THOUGH IT IS COMMENTED OUT FOR
50 ;VERSION 1. ANY LINE BEGINNING WITH ";VER2" SHOULD BE UNCOMMENTED. SEE
51 ;DOCUMENTATION. -RWY
52
53 ;REGISTER USAGE:
54 ; R0 TEMPORARY, I/O STRING POINTER
55 ; R1 TEMPORARY, USED BY I/O ROUTINES TO PASS CHARACTERS
56 ; R2 TEMPORARY, USED TO PASS ARGUMENTS BETWEEN FUNCTIONS
57 ; R3 LESS TEMPORARY - LOW LEVEL ROUTINES SHOULD NOT USE THIS
58 ; R4 STATUS WORD (IF NEEDED LATER, THIS CAN BE PUT IN MEMORY)
59 ; BIT0: CALLED FROM CHAIN FILE
60 ; BIT1: SYSTEM HAS NO PRINTER
61 ; BIT2: SET - 2K ROM, CLEAR - 8K ROM
62 ; BIT7: DELETE KEY WAS STRUCK
63 ; BIT15: XOFF HAS BEEN SENT
64 ; R5 OUTPUT DEVICE BUFFER ADDRESS
65
66
```

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

.TITLE EAROM BLASTER UTILITY
;.DSABL GBL
.NLIST CND,TOC,BEX,ME,MD
;.ENABL ABS,AMA,LC
.SBTTL UTILITY REVISION HISTORY

```
***
:
: REVISION      DATE          AUTHOR
: 0.0           JAN 84        NESTOR HERNANDEZ
:
: REVISION HISTORY:
: REVISION      CHANGE
: 1.0           INITIAL RELEASE VERSION.
:                REWRITTEN ALMOST COMPLETELY BY
:                RUSSELL YOUNG FOR APRIL RELEASE.
:
:---
```

.LIST MD

PROGRAM DEFINED MACROS

```
99          .SBTTL PROGRAM DEFINED MACROS
100
101
102          ;JUMP AND RETURN
103          .MACRO JRS ADDRESS
104             JMP ADDRESS
105          .ENDM
106
107          ;ALIGN TO BYTE 2 OF AN 8 WORD BLOCK
108          ;USED TO EASE ERROR RECOVERY ON INPUT PARAMETERS
109          .MACRO ALIGN2
110             .IF NE <<.-BEGIN>&6>-2
111                .*=.2
112                ALIGN2
113             .ENDC
114          .ENDM
115
116          ;TYPE A MESSAGE
117          .MACRO .TYPMSG ARG
118             .NARG ARGS
119             .IF NE ARGS
120                MOV ARG,RO
121             .ENDC
122             JSR PC,.TYPMSG
123          .ENDM
```

PROGRAM DEFINED MACROS

125
126

.NLIST ME,MD
.LIST MC

INITIALIZATION CODE

```

128          .SBTTL  INITIALIZATION CODE
129
130
131          ;**
132          ;
133          ;  INITIALIZATION ROUTINE
134          ;
135          ;  ROUTINE TO:
136          ;
137          ;  SETUP THE DRVCOM AND OTHER TRAP VECTORS
138          ;  SETUP THE DISPLAY & SOFTWARE SWITCH REGISTERS
139          ;  SETUP I/O BUFFER
140          ;  SIZE SYSTEM MEMORY
141          ;  INITIALIZE PROGRAM STACK
142          ;  SAVE MONITOR STACK POINTER
143          ;  CLEAR THE EAROM IMAGE STORAGE AREA
144          ;  DETERMINE EAROM SIZE AND LAST ADDRESS
145          ;  SEARCH FOR THE ADDRESS OF UFD HEADER
146          ;  SEARCH FOR UFD EXPANSION LIMIT
147          ;
148          ;--
149          ;PROGRAM SHOULD BE LINKED SO BEGIN FALLS AT ADDRESS 200 SO 200G WILL
150          ; DO A RESTART. MORE IMPORTANTLY, BEGIN -->MUST<-- FALL ON AN 8 BYTE
151          ; BOUNDARY OR THE PROGRAM WILL NOT WORK. REFER TO THE COMMENTS AROUND
152          ; UPDRAM AND THE MACRO ALIGN2.
153
154          000000          BEGIN:          ; BEGININIT
155          ;              .WORD          0
156          000000 005767 003340          TST          INISTK          ;REENTRY?
157          000004 001403          BEQ          5$          ;NO - PROCEED
158          000006 016706 003332          MOV          INISTK,SP      ;YES - RESTORE STACK, SKIP INITIALIZATION
159          000012 000410          BR          7$
160
161          000014 010667 003324          5$:          MOV          SP,INISTK      ;SET UP FOR A 200G TO RESTART
162          000020 012737 000200 000024  MOV          #200,@#24      ;SET UP FOR APT
163          000026 012737 003370 000044  MOV          #APTINF,@#44   ;DUMMY INFORMATION FOR APT
164
165          000034 012704 000001          7$:          MOV          #BIT0,R4      ;REENTER HERE ON ^C OR 200G
166          000040 013746 000042          MOV          @#42,-(SP)
167          000044 001004          BNE          10$
168          000046 012604          MOV          (SP),R4        ;FIX STACK, CLEAR STATUS
169          000050 023716 000046          CMP          @#46,(SP)
170          000054 001540          BEQ          OUT$
171
172          000056 013746 000034          10$:         MOV          @#34,-(SP)      ;SAVE MONITER TRAPS AND USE MY OWN ROUTINE
173          ;VER2 MOV          #TRPHAN,@#34      ;:(IS THIS NEEDED?)
174          000062 013746 177520          MOV          @#BCSR,-(SP)   ; SAVE OLD BCSR VALUE
175          000066 042737 000100 177520  BIC          #BIT6,@#BCSR   ;ENABLE BOOT ROM
176          000074 052737 000040 177520  BIS          #BIT5,@#BCSR
177
178          000102 012737 000102 000100  MOV          #102,@#100     ; : SET UP LINE TIME CLOCK VECTOR AT 100
179          000110 012737 000002 000102  MOV          #2,@#102       ; : RETURN FROM INTERRUPT
180
181          000116 012703 000006          MOV          #6,R3
182          000122 011346          MOV          (R3),-(SP)
183          000124 013713 177776          MOV          @#177776,(R3)  ;DON'T LOSE CORRECT SPACE
184          000130 014346          MOV          -(R3),-(SP)

```

INITIALIZATION CODE

```

185 000132 012713 000152'      MOV    #30$(R3)      ;SIZE MEMORY TO FIND WHERE MONITOR IS
186 000136 005000                CLR    R0
187 000140 062700 020000      20$:  ADD    #20000,R0
188 000144 001405                BEQ    40$
189 000146 005710                TST    (R0)
190 000150 000773                BR     20$
191 000152 022626      30$:  CMP    (SP)+,(SP)+  ;GET HERE THROUGH BUS TIME OUT - FIX STACK
192 000154 062700 020000      ADD    #20000,R0  ;EXTRA 20000 ADDED HERE TOO
193 000160 162700 030000      40$:  SUB    #30000,R0  ;COMPENSATE FOR EXTRA ADD IN EITHER CASE
194 000164 010067 007156      MOV    R0,MONADR
195
196 000170 012713 000202'      MOV    #50$(R3)
197 000174 005737 000000      TST    @#PRTCSR
198 000200 000403                BR     60$
199 000202 022626      50$:  CMP    (SP)+,(SP)+  ;IF HERE, NO PRINTER ON BUS, SO USE CRT
200 000204 052704 000002      BIS    #BIT1,R4
201 000210                .TYPMSG #HELLO      ; : PRINT UTILITY IDENTIFICATION
202 000220 004767 003014      JSR    PC,REAROM  ; EAROM DATA INTO SYSTEM MEMORY
203 000224 012623                MOV    (SP)+,(R3)+  ;RESTORE TRAP 4
204 000226 012613                MOV    (SP)+,(R3)+
205 000230 103411                BCS    110$        ;ROM NONEXISTANT, PRINT ERROR AND EXIT
206 000232 012702 000410'      MOV    #MAINM,R2  ;LOAD TABLE AND FALL INTO MENU
207 000236 004767 000024      JSR    PC,MENU
208 000242 012637 177520      100$: MOV    (SP)+,@#BCSR  ;RESTORE OLD BCSR VALUE
209 000246 012637 000034      MOV    (SP)+,@#34  ;RESTORE MONITER TRAP HANDLER
210 000252 000207      RTS    PC
211
212 000254                110$: .TYPMSG #NOROM
213 000264 000766      BR     100$      ;RETURN TO MONITOR
214

```


INITIALIZATION CODE

```

216 ;MENU - GIVEN A STRUCTURE POINTER IN R2, WILL PRINT THE MENU, READ THE
217 ; CHOICE, AND PERFORM IT. IT USES R0 AND R3
218 000266 010246 MENU: MOV R2,-(SP)
219 000270 011603 PAINT: MOV (SP),R3 ;RELOAD MENU POINTER
220 000272 .TYPMSG (R3)+ ;PRINT THE MENU
221 000300 012700 012707' MOV #PROMPT,R0
222 000304 004767 002140 JSR PC,GETCOM ;GET THE COMMAND
223 000310 012301 MOV (R3)+,R1 ;R1 NOW HOLDS LEGAL COMMAND POINTERS
224 000312 005300 DEC R0 ;R0 IS THE NUMBER OF CHARACTERS
225 000314 002407 BLT 20$ ;DEFAULT RETURN
226 000316 003011 BGT ERROR1 ;MORE THAN 1 CHARACTER IS ILLEGAL
227 000320 005723 10$: TST (R3)+ ;INCREMENT FUNCTION POINTER
228 000322 105711 TSTB (R1) ;CHECK IF DONE
229 000324 001403 BEQ 20$
230 000326 122167 003016 CMPB (R1)+,INSTR ;CHECK NEXT POSSIBLE CHOICE
231 000332 001372 BNE 10$
232 000334 005001 20$: CLR R1 ;SET UP FOR HELP REQUEST
233 000336 004733 JSR PC,@(R3)+ ;DO FUNCTION
234 000340 000753 BR PAINT ;REDRAW SCREEN
235
236 000342 012746 000270' ERROR1: MOV #PAINT,-(SP) ;SET UP FOR RTS
237 000346 012700 012414' ERROR: MOV #ILLCOM,R0 ;ILLEGAL COMMAND - PRINT ERROR, REDRAW MENU
238 000352 JRS GETCOM ;GET <CR> BEFORE PAINTING MENU
239
240

```

INITIALIZATION CODE

```

242          ;MENU ROUTINES - LOAD MENU STRUCTURE POINTER IN R2 AND GO TO COMMON CODE
243
244          ;VER2M.U:      MOV      #UPDP,R2          ;UPDATE PERIPHERALS
245          ;VER2      JSR      PC,MENU
246          ;PERIPHERAL PROCESSING GOES HERE
247 000356  REPLOTT:      ;CONVENIENT RTS FOR JUMPING TO
248 000356  OUT$:      RTS      PC
249
250 000360  012702  000440'  M.S:      MOV      #HARDM,R2          ;CREATE SYS COM FILE FROM ROM MENU
251 000364  000740          BR      MENU
252
253 000366  MS.H:      .TYPMSG #MSG4          ;PRINT THE HELP MESSAGE
254 000376          JRS      RETCOM
255
256 000402  012702  000474'  M.H:      MOV      #HELPM,R2          ;HELP MENU
257 000406  000727          BR      MENU
258
259

```

INITIALIZATION CODE

```

261 ;MENU STRUCTURES:
262 ; WORD 0: ADDRESS OF ASCII STRING OF MENU
263 ; WORD 1: ADDRESS OF LEGAL CHARACTER COMMANDS
264 ; WORD 2: ADDRESS OF DEFAULT FUNCTION
265 ; WORD 3-N: FUNCTION ADDRESSES IN ORDER
266 ; BYTE 2N+: NULL TRAILED LEGAL CHARACTORS
267
268 ;MAIN MENU
269 000410 003404' MAINM: .WORD $MENU ;ASCII STRING FOR MAIN MENU
270 000412 000432' .WORD M.CHR ;ADDRESS OF LEGAL COMMANDS
271 ; FUNCTIONS
272 000414 000646' .WORD M.E ;DEFAULT - EXIT
273 ;VER2 .WORD M.F ;CREATE SYS CONFIG FILE FROM ROM
274 ;VER2 .WORD M.C ;COPY TO ROM FROM SYS CONFIG FILE
275 ;VER2 .WORD M.U ;UPDATE SYS MEMORIES, PERIPHERALS
276 000416 000360' .WORD M.S ;UPDATE SYS HARDWARE
277 000420 000560' .WORD M.D ;PRINT SYSTEM DESCRIPTION
278 000422 000356' .WORD REPLOT ;REDISPLAY MENU
279 000424 000402' .WORD M.H ;HELP
280 000426 000646' .WORD M.E ;EXIT
281 000430 000346' .WORD ERROR ;NO MATCH
282 ;VER2 M.CHR: .ASCIZ /FCUSDRHE/
283 000432 123 104 122 M.CHR: .ASCIZ /SDRHE/
284 .EVEN
285
286 ;UPDATE HARDWARE MENU
287 000440 004001' HARDM: .WORD $HMENU
288 000442 000464' .WORD MS.CHR
289 000444 002762' 000666' 000666' .WORD EXIT,MS.C,MS.U,MS.B,MS.S,MS.H,EXIT,ERROR
290 000464 103 125 102 MS.CHR: .ASCIZ /CUBSHR/
291 .EVEN
292
293 000474 004562' HELPM: .WORD $HELP
294 000476 000512' .WORD MH.CHR
295 ;VER2 .WORD EXIT,HMSG1,HMSG2,HMSG3,HMSG4,HMSG5,HMSG6,ERROR
296 ;VER2 MH.CHR: .ASCIZ /FCUSDE/
297 000500 002762' 000614' 000620' .WORD EXIT,HMSG4,HMSG5,HMSG6,ERROR
298 000512 123 104 105 MH.CHR: .ASCIZ /SDE/
299 .EVEN
300
301 ;VER2UPDP: .WORD $MORP ;UPDATE MEMORY OR PERIPHERALS
302 ;VER2 .WORD MU.CHR
303 ;VER2 .WORD EXIT,YUP,NOPE,ERROR
304 ;VER2MU.CHR: .ASCIZ /MP/
305 .EVEN
306
307 ;VER2SUREMU: .WORD $SURE ;"ARE YOU SURE?"
308 ;VER2 .WORD YN
309 ;VER2 .WORD NOPE,YUP,NOPE,ERROR
310 ;VER2 ;USES SAME CHARACTERS AS YORN
311
312 000516 012333' YORN: .WORD $TYASK ;PRINT ON HARD COPY?
313 000520 000532' .WORD YN
314 000522 002762' 002762' 002770' .WORD YUP,YUP,NOPE,ERROR
315 000532 131 116 000 YN: .ASCIZ /YN/
316 .EVEN
317

```

INITIALIZATION CODE

```
318 000536 004326'          EMENU: .WORD $EMENU
319 000540 000554'          .WORD RWE
320 000542 002770' 002776' 003170' .WORD NOPE,WRTROM,LEAVE,NOPE,ERROR
321 000554   127   105   122 RWE:  .ASCIZ /WER/
322
```

INITIALIZATION CODE

```

324          ;      FUNCTIONS
325          ;      -----
326          ; FROM MAIN MENU
327
328          ;VER2M.F:      MOV      #DUMP,-(SP)      ;LOAD FILE TO XXDP MEDIUM FROM ROM
329          ;VER2      MOV      #WRITE,-(SP)
330          ;VER2      BR       FILCOM
331          ;VER2M.C:      MOV      #LOAD,-(SP)      ;WRITE ROM FROM XXDP MEDIUM
332          ;VER2      MOV      #READ,-(SP)
333          ;VER2      BR       FILCOM
334          ;VER2
335          ;VER2M.FERR:    .TYPMSG #BADFIL
336          ;VER2FILCOM:    ;COMMON READ/WRITE CODE
337          ;VER2      MOV      #GETFIL,R0
338          ;VER2      JSR      PC,GETCOM
339          ;VER2      BEQ      98$
340          ;VER2      JSR      PC,CHKFIL
341          ;VER2      BCS      M.FERR
342          ;VER2
343          ;VER2      .TYPMSG (SP)+
344          ;VER2      .TYPMSG #INSTR
345          ;VER2      MOV      #SUREMU,R2
346          ;VER2      JSR      PC,MENU
347          ;VER2      BCS      99$
348          ;VER2      MOV      (SP),R0
349          ;VER2      MOV      R4,(SP)
350          ;VER2      JSR      PC,(R0)
351          ;VER2      MOV      (SP)+,R4
352          ;VER2      .TYPMSG
353          ;VER2      JSR      PC,RETCOM
354          ;VER2      BR       100$
355          ;VER2
356          ;VER298$:      TST      (SP)+
357          ;VER299$:      TST      (SP)+
358          ;VER2100$:    RTS      PC
359
360 000560 012705 000700' M.D:      MOV      #MSTAB,R5
361 000564 012502          10$:      MOV      (R5)+,R2
362 000566 001403          ;VER2      BEQ      100$
363 000570 004767 000600'          ;VER2      JSR      PC,DESCRP
364 000574 000773          ;VER2      BR       10$
365
366 000576 000207          100$:    RTS      PC
367
368          ;MAIN MENU HELP FUNCTION
369 000600 062701 177525 HMSG1:  ADD      #HMSG1-HMSG2,R1      ;LOAD R1 WITH ADDRESS OF HELP MESSAGE
370 000604 062701 177527 HMSG2:  ADD      #HMSG2-HMSG3,R1      ; TO PRINT
371 000610 062701 177334 HMSG3:  ADD      #HMSG3-HMSG4,R1
372 000614 062701 177412 HMSG4:  ADD      #HMSG4-HMSG5,R1
373 000620 062701 177651 HMSG5:  ADD      #HMSG5-HMSG6,R1
374 000624 062701 012035' HMSG6:  ADD      #HMSG6,R1
375 000630          ;VER2      .TYPMSG R1
376 000636 004767 001602 10$:      JSR      PC,RETCOM      ;TYP <RET> TO RETURN
377 000642 001375          ;VER2      BNE      10$      ;IF ANY OTHER CHAR. TYPED TRY AGAIN
378 000644 000207          ;VER2      RTS      PC
379
380 000646 012702 000536' M.E:      MOV      #EMENU,R2

```

INITIALIZATION CODE

```

381 000652 004767 177410      JSR    PC,MENU
382 000656 103402             BCS    100$
383 000660 000167 002076      JMP    EXIT                ;RETURN TO EXIT PROGRAM
384 000664 000207             RTS    PC                ;IF R RETURN TO MENU
385
386 000666                   MS.C:
387 000666                   MS.U:
388 000666                   MS.B:
389 000666 062703 000230      MS.S:  ADD    #MSTAB-HARDM-10,R3    ;BUILD OFFSET INTO DESCRIPTOR TABLE
390 000672 011302             MOV    (R3),R2
391 000674                   JRS    UPDRAM
392
393 000700 000742' 001072' 001132' MSTAB: .WORD  CPUDT,UBADT,BOTDT,SWIDT,0

```

INITIALIZATION CODE

```

395
396 ;INPUT AND OUTPUT MAPPING ROUTINES FOR DISPLAYING AND MODIFYING DATA
397 000712 OUTMAP: ;OUTPUT MAPPING ROUTINES
398 000712 001570' .WORD OTYP0 ;TYPE 0 - BIT INFORMATION, 0 <= N <= 7
399 000714 001646' .WORD OTYP2 ;TYPE 2 - PRINT DECIMAL NUMBER IN R3
400 000716 001772' .WORD OTYP4 ;TYPE 4 - 8 BIT OCTAL
401 000720 002002' .WORD OTYP6 ;TYPE 6 - 16 BIT OCTAL
402 000722 002100' .WORD OTYP10 ;TYPE 10 - GET TWO ASCII CHARACTERS
403
404 000724 INMAP: ;INPUT MAPPING ROUTINES
405 000724 002144' .WORD ITYP0 ;TYPE 0 - BIT INFORMATION, 0 <= N <= 7
406 000726 002212' .WORD ITYP2 ;TYPE 2 - PRINT DECIMAL NUMBER IN R3
407 000730 002314' .WORD ITYP4 ;TYPE 4 - 8 BIT OCTAL
408 000732 002334' .WORD ITYP6 ;TYPE 6 - 16 BIT OCTAL
409 000734 002352' .WORD ITYP10 ;PRINT TWO ASCII CHARACTERS
410
411 ;THESE ARE THE DESCRIPTOR TABLES. THE FORMAT IS AS FOLLOWS:
412 ;IT CONSISTS OF HEADER INFORMATION AND 8 BYTES FIELD DESCRIPTORS. THE
413 ; FIELD DESCRIPTORS DIFFER DEPENDING ON THE TYPE OF INFORMATION THEY
414 ; ACCESS, BUT THEY MUST BE 8 BYTES LONG.
415
416 :
417 : BYTE 0: NUMBER OF LINES IN TABLE
418 : BYTE 1: WIDTH OF TABLE HEADER
419 : WORD 1: ASCIZ TABLE NAME
420 : WORD 2: HEADER ASCII STRING
421 :
422 : WORD N: PROMPT FOR INPUT OF FOLLOWING FIELD
423 : WORD N+2: ADDRESS IN UPDATE OF BYTE
424 : THESE LAST FOUR BYTES VARY DEPENDING ON THE FIELD TYPE
425 :
426 : BYTE N+4: TYPE 0 - BIT FIELDS
427 : BYTE N+5: 0 (DATA TYPE OFFSET INTO I/O FUNCTION TABLES)
428 : BYTE N+6: MASK THE INTERESTING BITS
429 : BYTE N+7: BITS TO SHIFT TO REACH THE MASK
430 :
431 : BYTE N+4: TYPE 2 - DECIMAL INPUT
432 : BYTE N+5: 2 (DATA TYPE OFFSET INTO I/O FUNCTION TABLES)
433 : BYTE N+6: NUMBER OF BYTES TO PRINT
434 : BYTE N+7: MAXIMUM VALUE
435 :
436 : BYTE N+4: TYPE 4 - INPUT OCTAL BYTE
437 : BYTE N+5: 4 (DATA TYPE OFFSET INTO I/O FUNCTION TABLES)
438 : BYTE N+6: 2 (NUMBER OF BITS TO ROTATE IN TO FIRST DIGIT)
439 : BYTE N+7: UNUSED
440 :
441 : BYTE N+4: TYPE 6 - INPUT OCTAL WORD
442 : BYTE N+5: 6 (DATA TYPE OFFSET INTO I/O FUNCTION TABLES)
443 : BYTE N+6: 1 (NUMBER OF BITS TO ROTATE IN TO FIRST DIGIT)
444 : BYTE N+7: UNUSED
445 :
446 : BYTE N+4: TYPE 10 - INPUT ASCII STRING
447 : BYTE N+5: 10 (DATA TYPE OFFSET INTO I/O FUNCTION TABLES)
448 : BYTE N+6: MAXIMUM NUMBER OF CHARS
449 : BYTE N+7: UNUSED
450
451 000736 ALIGN2 ;SET ON RIGHT BOUNDARY

```

INITIALIZATION CODE

```

452
453 000742 001 107 CPUDT: .BYTE 1,71.
454 000744 005077' .WORD CPUIIT
455 000746 005131' .WORD CPUDES ;TEXT LENGTH, HEADING
456
457 000750 006353' .WORD CPUP0
458 000752 000000G .WORD UPDATE.0
459 000754 000 100 006 .BYTE 0,BIT6,6 ;IGNORE BATTERY STATUS
460 .EVEN
461 000760 006451' .WORD CPUP1
462 000762 000001G .WORD UPDATE.1
463 000764 000 300 006 .BYTE 0,BIT7:BIT6,6 ;POWER UP MODE
464 .EVEN
465 000770 006547' .WORD CPUP2
466 000772 000001G .WORD UPDATE.1
467 000774 000 060 004 .BYTE 0,BIT5:BIT4,4 ;REBOOT MODE
468 .EVEN
469 001000 006645' .WORD CPUP3
470 001002 000001G .WORD UPDATE.1
471 001004 000 007 000 .BYTE 0,BIT2:BIT1:BIT0,0 ;MASTER BUS GRANT
472 .EVEN
473 001010 006743' .WORD CPUP4
474 001012 000001G .WORD UPDATE.1
475 001014 000 010 003 .BYTE 0,BIT3,3 ;HALT/TRAP OPTION
476 .EVEN
477 001020 007041' .WORD CPUP5
478 001022 000000G .WORD UPDATE.0
479 001024 000 002 001 .BYTE 0,BIT1,1 ;HALT ON BREAK
480 .EVEN
481 001030 007137' .WORD CPUP6
482 001032 000000G .WORD UPDATE.0
483 001034 000 020 004 .BYTE 0,BIT4,4 ;LINE CLOCK STATUS
484 .EVEN
485 001040 007235' .WORD CPUP7
486 001042 000000G .WORD UPDATE.0
487 001044 000 040 005 .BYTE 0,BIT5,5 ;LINE CLOCK INTERRUPT
488 .EVEN
489 001050 007333' .WORD CPUP8
490 001052 000000G .WORD UPDATE.0
491 001054 000 014 002 .BYTE 0,BIT3:BIT2,2 ;LINE CLOCK SOURCE
492 .EVEN
493 001060 007431' .WORD CPUP9
494 001062 000003G .WORD UPDATE.3
495 001064 000 001 000 .BYTE 0,BIT0,0 ;MAIN MEMORY TEST
496 .EVEN
497 001070 000000 .WORD 0
498
499 001072 001 026 UBADT: .BYTE 1,22.
500 001074 005467' .WORD UBATIT ;TABLE HEADER
501 001076 005521' .WORD UBADES
502
503 001100 006130' .WORD UBAPO ;UNIBUS CACHE
504 001102 000002G .WORD UPDATE.2
505 001104 000 100 006 .BYTE 0,BIT6,6
506 .EVEN
507 001110 006211' .WORD UBAP1 ;18 BIT MODE
508 001112 000002G .WORD UPDATE.2

```


INITIALIZATION CODE

509	001114	000	040	005	.BYTE	0,BIT5,5	
510					.EVEN		
511	001120	006272'			.WORD	UBAP2	;UNIBUS MEMORY TEST
512	001122	000004G			.WORD	UPDATE+4	
513	001124	000	002	001	.BYTE	0,BIT1,1	
514					.EVEN		
515							
516	001130	000000			.WORD	0	;END OF TABLE
517							
518	001132	011	045		BOTDT: .BYTE	9..37.	
519	001134	005634'			.WORD	BOTTIT	
520	001136	005677'			.WORD	BOTDES	
521							
522	001140	007527'			.WORD	BOTPO	
523	001142	000000			.WORD	0	;DON'T CARE ABOUT ADDRESS FOR TYPE 2
524	001144	002	004	011	.BYTE	2,4,11,6	
525							
526	001150	007601'			.WORD	BOTP1	;DEVICE MNEMONIC
527	001152	000036G			.WORD	UPDATE+36	
528	001154	010	002	000	.BYTE	10,2,0,0	;MAKE IT 8 BYTES
529							
530	001160	007654'			.WORD	BOTP2	;PHYSICAL UNIT #
531	001162	000040G			.WORD	UPDATE+40	
532	001164	004	002	000	.BYTE	4,2,0,0	
533							
534	001170	007727'			.WORD	BOTP3	;LOGICAL UNIT NUMBER
535	001172	000041G			.WORD	UPDATE+41	
536	001174	004	002	000	.BYTE	4,2,0,0	
537							
538	001200	010002'			.WORD	BOTP4	;CSR
539	001202	000042G			.WORD	UPDATE+42	
540	001204	006	001	000	.BYTE	6,1,0,0	
541							
542	001210	000006			.WORD	6	;END OF TABLE
543							
544	001212	006	024		SWIDT: .BYTE	6,20.	
545	001214	006016'			.WORD	SWITIT	
546	001216	006051'			.WORD	SWIDES	
547							
548	001220	010055'			.WORD	SWIPO	;SWITCH NUMBER TO MODIFY
549	001222	000000			.WORD	0	
550	001224	002	002	006	.BYTE	2,2,6,3	
551							
552	001230	007601'			.WORD	BOTP1	;NEW DEVICE MNEMONIC
553	001232	000124G			.WORD	UPDATE+124	
554	001234	010	002	000	.BYTE	10,2,0,0	
555							
556	001240	007727'			.WORD	BOTP3	;LOGICAL UNIT NUMBER
557	001242	000126G			.WORD	UPDATE+126	
558	001244	004	002	000	.BYTE	4,2,0,0	
559							
560	001250	000003			.WORD	3	

INITIALIZATION CODE

```

562
563
564
565
566
567 001252 010246
568 001254 004767 000114
569 001260 005067 000106
570
571 001264 011602
572 001266 005742
573
574 001270 062702 000010
575 001274 012200
576 001276 032700 177740
577 001302 001426
578 001304 010246
579 001306 004767 001136
580 001312 012602
581 001314 005700
582 001316 001410
583 001320 012203
584 001322 066703 000044
585 001326 112201
586 001330 010546
587 001332 004771 000724
588 001336 012605
589 001340 042702 000007
590 001344 103351
591 001346
592 001356 000746
593
594 001360 012602
595 001362 004767 000006
596 001366
597 001372 000000
598
599
600
601 001374 010546
602 001376 112205
603 001400 112200
604 001402 010046
605 001404
606 001412 012701 000055
607 001416 011600
608 001420 004767 001234
609 001424 077003
610 001426
611 001434 012701 000055
612 001440 012767 000001 177724
613
614 001446 011600
615 001450 004767 001204
616 001454 077003
617 001456
618

;MAIN ROUTINES TO DISPLAY AND UPDATE A SET OF DATA
;UPDATE A SET OF DATA
;FRONT OF THE DESCRIPTOR TABLE IS PASSED IN R2
UPDRAM: MOV R2,-(SP) ;SAVE DESCRIPTOR TABLE POINTER
        JSR PC,DESCRP ;PRINT OUT CURRENT VALUES
        CLR WHICH ;INIT TO FRONT OF TABLE

        MOV (SP),R2 ;RESTORE POINTER
        TST -(R2) ;6 SHOULD BE ADDED THE FIRST TIME - SUB 2 HERE

10$: ADD #10,R2 ;NEXT ENTRY, OVERLOOK HEADER THE FIRST TIME
20$: MOV (R2)+,R0 ;OUTPUT PROMPT FOR INPUT
        BIT #177740,R0 ;CHECK FOR END
        BEQ 100$ ;EQUAL SIGNALS END
        MOV R2,-(SP)
        JSR PC,GETCOM ;GET COMMAND
        MOV (SP)+,R2
        TST R0 ;TEST FOR DEFAULT
        BEQ 30$
        MOV (R2)+,R3 ;SET UP INPUT ADDRESS
        ADD WHICH,R3 ;SET UP IN ITYP2
        MOVB (R2)+,R1
        MOV R5,-(SP)
        JSR PC,@INMAP(R1) ;DO THIS TYPE OF INPUT
        MOV (SP)+,R5
30$: BIC #7,R2 ;RESTORE R2 TO PREVIOUS ENTRY
        BCC 10$ ;GOOD RETURN, GET NEXT
        .TYPMSG #ERRIN ;BAD INPUT
        BR 20$ ;TRY IT AGAIN

100$: MOV (SP)+,R2 ;PRINT OUT UPDATED TABLE AND RETURN
        JSR PC,DESCRP
        JRS RETCOM
WHICH: .WORD

;PRINT OUT A DESCRIPTIVE HEADING
;DESCRIPTOR POINTER IN R2
DESCRP: MOV R5,-(SP)
        MOVB (R2)+,R5 ;REPEAT COUNT
        MOVB (R2)+,R0
        MOV R0,-(SP) ;WIDTH OF HEADER
        .TYPMSG (R2)+ ;PRINT THE HEADER
        MOV #'-',R1
        MOV (SP),R0
10$: JSR PC,TTY
        SOB R0,10$
        .TYPMSG (R2)+
        MOV #'-',R1 ;CLOSING ----S
        MOV #1,WHICH ;INITIALIZE WITH #1

20$: MOV (SP),R0
        JSR PC,TTY
        SOB R0,20$
        .TYPMSG #CRLF

;PROCESS THE TABLES

```

INITIALIZATION CODE

```

619 001466 010246          MOV      R2,-(SP)      ;SAVE FRONT OF DESCRIPTER COUNTER FOR REPEATS
620 001470 005046          CLR      -(SP)        ;INCREMENT TABLE POINTER
621 001472 016602 000002   30$:    MOV      2(SP),R2  ;RESTORE DESCRIPTER POINTER
622 001476 011603          40$:    MOV      (SP),R3    ;SET UP OFFSET INTO TABLE
623 001500 032722 177740   BIT      @177740,(R2)+ ;THIS ENTRY IS EITHER EOT OR ASCII PROMPT
624 001504 001410          BEQ     100$          ;EOT - FINISH AND TRY NEXT
625 001506 062203          ADD     (R2)+,R3     ;POINT INTO TABLE
626 001510 112200          MOVB   (R2)+,R0
627 001512 004770 000712'  JSR     PC,@OUTMAP(R0)
628 001516 052702 000007   BIS     @7,R2        ;POINT TO NEXT ENTRY
629 001522 005202          INC     R2
630 001524 000764          BR     40$
631
632 001526          100$:   .TYPMSG @VCRLF
633 001536 064216          ADD     -(R2),(SP)   ;OFFSET TO NEXT
634 001540 005267 177626   INC     WHICH        ;NEXT IN NUMERIC ORDER
635 001544 077526          SOB    R5,30$
636
637 001546 022626          CMP     (SP)+,(SP)+ ;BUMP OFF TEMP STORAGE
638 001550 012600          MOV     (SP)+,R0
639 001552 012701 000055   MOV     #'-,R1       ;PRINT FINAL LINE
640 001556 004767 001076   110$:  JSR     PC,TTY
641 001562 077003          SOB    R0,110$
642 001564 012605          MOV     (SP)+,R5
643 001566 000207          RTS     PC
644

```

INITIALIZATION CODE

```

646 ;SEPARATE ROUTINES TO HANDLE THE DIFFERENT FORMS THE DATA IS DISPLAYED
647 ; AND ENTERED IN
648
649 ;PRINT ROUTINES
650
651 ;DATA TYPE 0 - THE OUTPUT WILL BE A SINGLE CHARACTER, 0 <= X <= 7
652 ;THE ADDRESS OF THE DATA IS IN R3, R2 POINTS AT BYTE 2 OF THE DATA DESCRIPTOR
653 001570 OTYPO: .TYPMSG @LBL5 ;LEADING 4 BLANKS
654 001600 111301 MOV (R3),R1
655 001602 042701 177400 BIC #177400,R1 ;CLEAR POSSIBLE SIGN BITS
656 001606 112200 MOV (R2)+,R0 ;LOAD MASK BITS
657 001610 005100 COM R0
658 001612 040001 BIC R0,R1
659 001614 112200 MOV (R2)+,R0
660 001616 005100 COM R0 ;SET UP NEGATIVE FOR RIGHT SHIFT
661 001620 005200 INC R0
662 001622 072100 ASH R0,R1 ;SHIFT BITS TO RIGHT
663 001624 062701 000060 ADD #0,R1
664 001630 004767 001024 JSR PC,TTY ;PRINT THE NUMBER
665 001634 .TYPMSG @BL4 ;TRAILING BLANKS
666 001644 000207 RTS PC
667
668 ;OUTPUT A DECIMAL NUMBER
669 001646 016703 177520 OTYP2: MOV WHICH,R3
670
671 001652 012701 000174 MOV #' ),R1 ;HERE IS A GENERAL OUTPUT DECIMAL FUNCTION
672 001656 004767 000776 JSR PC,TTY
673 001662 012701 000040 MOV #' ,R1
674 001666 004767 000766 JSR PC,TTY
675
676 001672 111200 MOV (R2),R0 ;NUMBER OF BYTES TO PRINT * 2
677 001674 062700 001760' ADD @DECTAB,R0 ;DECIMAL TABLE
678 001700 012746 000017 MOV #17,-(SP) ;NO NONZERO DIGITS PRINTED
679
680 001704 012701 000057 10$: MOV #'0-1,R1 ;BUILD A DIGIT
681 001710 005201 20$: INC R1
682 001712 161003 SUB (R0),R3
683 001714 002375 BGE 20$
684
685 001716 031601 BIT (SP),R1 ;SUPPRESS LEADING 0S
686 001720 001003 BNE 30$
687 001722 012701 000040 MOV #' ,R1
688 001726 000402 BR 40$
689
690 001730 012716 177777 30$: MOV #-1,(SP) ;0 FILL INSTEAD OF BLANK
691 001734 004767 000720 40$: JSR PC,TTY ;PRINT IT
692 001740 061003 ADD (R0),R3 ;RESTORE EXTRA SUBTRACTION
693 001742 005740 TST -(R0) ;GET NEXT LOWER POWER OF 10
694 001744 001357 BNE 10$ ;IF NOT DONE, DO NEXT DIGIT
695
696 001746 005726 TST (SP), ;LOSE TEMPORARY VARIABLE
697 001750 012701 000040 MOV #' ,R1 ;TRAILING BLANK
698 001754 JRS TTY
699
700 001760 000000 000001 000012 DECTAB: .WORD 0..1..10..100..1000.
701
702 ;OUTPUT 16 BIT OCTAL NUMBER

```

INITIALIZATION CODE

```

703 001772 005300      OTYP4:  DEC    R0          ;NUMBER OF CHARS IN R0
704 001774 111303      MOVB   (R3),R3        ;NUM IN R3 HIGH BYTE
705 001776 000303      SWAB   R3
706 002000 000406      BR     OCTOUT
707
708 002002 005001      OTYP6:  CLR    R1
709 002004 152301      BISB  (R3)+,R1
710 002006 000301      SWAB  R1
711 002010 151301      BISB  (R3),R1
712 002012 010103      MOV   R1,R3
713 002014 000303      SWAB  R3
714
715 002016 012701 000174  OCTOUT: MOV   #' ),R1
716 002022 004767 000632  JSR   PC,TTY
717 002026 012701 000040  MOV   #' ,R1
718 002032 004767 000622  JSR   PC,TTY
719 002036 112201      MOVB  (R2)+,R1        ;HOW MANY BITS IN FIRST DIGIT?
720 002040 010246      MOV   R2,-(SP)       ;ASHC NEEDS R-EVEN, R-ODD
721
722 002042 005002      10$:  CLR    R2          ;NO STRAY BITS
723 002044 073201      ASHC  R1,R2          ;SHIFT HIGH THREE BITS INTO R2
724 002046 010201      MOV   R2,R1         ;PRINT IT
725 002050 052701 000060  BIS   #'0,R1        ;MAKE ASCII
726 002054 004767 000600  JSR   PC,TTY
727 002060 012701 000003  MOV   #3,R1         ;SHIFT 3 ON SUBSEQUENT ATTEMPTS
728 002064 077012      SOB   R0,10$
729
730 002066 012602      MOV   (SP)+,R2       ;RESTORE R2 TO TABLE
731 002070 012701 000040  MOV   #' ,R1
732 002074      JRS   TTY
733
734      ;PRINT ASCII
735 002100      OTYP10: .TYPMSG @LBL5
736 002110 112200      MOVB  (R2)+,R0       ;TWO CHARACTERS
737 002112 112301      10$:  MOVB  (R3)+,R1
738 002114 022701 000040  CMP   #' ,R1
739 002120 002402      BLT   20$
740 002122 012701 000040  MOV   #' ,R1
741 002126 004767 000526  20$:  JSR   PC,TTY
742 002132 077011      SOB   R0,10$
743
744 002134 012700 013173'  MOV   #BL4,R0
745 002140      JRS   TYPMSG
746
747      ;INPUT FUNCTIONS
748      ;THESE FUNCTIONS DO THE INPUT FOR THE DIFFERENT DATA TYPES. THEY ARE ENTERED
749      ; WITH R3 POINTING TO THE ADDRESS IN UPDATE OF THE LOCATION FOR THE NEW DATA.
750      ; THE INPUT HAS ALREADY BEEN READ IN IN GETCOM, AND R0 IS LEFT HOLDING THE
751      ; NUMBER OF CHARACTERS IN THE ASCII STRING IN INSTR. R2 POINTS TO BYTE 5 OF THE
752      ; 8 BYTE FIELD DESCRIPTER. IF THE INPUT IS SUCCESSFUL THE FUNCTIONS SHOULD
753      ; RETURN C CLEAR AND R2 POINTING TO THE NEXT DESCRIPTER. IF THE INPUT IS
754      ; NOT CORRECT IT RETURNS C SET, AND R2 MUST BE POINTING SOMEWHERE IN THE
755      ; CURRENT DESCRIPTER - THEY ARE 8 BYTE ALIGNED SO A BIC #7,R2 WILL RESTORE
756      ; THE POINTER SO IT CAN BE REUSED.
757
758      ;INPUT SINGLE DIGIT OCTAL.
759

```

INITIALIZATION CODE

```

760 002144 112201          ITYP0:  MOVB   (R2)+,R1      ;HOLDS BIT MASK
761 002146 042701 177400  BIC    #177400,R1    ;CLEAR POSSIBLE SIGN BITS
762 002152 022700 000001  CMP    #1,R0
763 002156 103414          BCS    100$
764 002160 111200          MOVB   (R2),R0      ;HOLDS SHIFT
765 002162 016705 001162  MOV    INSTR,R5
766 002166 001410          BEQ    100$        ;RETURN GOOD ON DEFAULT (C BIT CLEAR FROM CMP)
767 002170 162705 000060  SUB    #'0,R5
768 002174 103405          BCS    100$
769
770 002176 072500          ASH    R0,R5
771 002200 020105          CMP    R1,R5      ;CHECK FOR LEGALITY
772 002202 103402          BCS    100$
773 002204 140113          BICB  R1,(R3)     ;CLEAR THE BITS
774 002206 150513          BISB  R5,(R3)     ;SET THE BITS
775 002210 000207          100$: RTS        PC
776
777          ;GET A DECIMAL NUMBER AND LOAD WHICH WITH THE OFFSET FOR THE DESIRED ENTRY
778 002212 012700 003350' ITYP2:  MOV    #INSTR,R0
779 002216 005005          CLR    R5        ;LOAD VALUE IN HERE
780 002220 005003          CLR    R3        ;USE FOR *10 MULTIPLICATION
781
782 002222 112001          10$:  MOVB   (R0)+,R1    ;EXAMINE NEXT DIGIT
783 002224 001415          BEQ    98$
784 002226 162701 000060  SUB    #'0,R1
785 002232 103427          BCS    100$      ;ISOLATE BITS, CHECK FOR RANGE
786 002234 022701 000011  CMP    #11,R1
787 002240 103424          BCS    100$
788 002242 006305          ASL    R5        ;MULTIPLY BY 10. 2X
789 002244 006305          ASL    R5        ;4X
790 002246 060305          ADD    R3,R5     ;5X
791 002250 006305          ASL    R5        ;10X
792 002252 060105          ADD    R1,R5     ;10X*Y
793 002254 010503          MOV    R5,R3
794 002256 000761          BR    10$
795
796 002260 005202          98$:  INC    R2        ;BUMP PAST BYTES TO PRINT
797 002262 122205          CMPB  (R2)+,R5   ;CHECK RANGE
798 002264 103412          BCS    100$
799 002266 020527 000001  CMP    R5,#1     ;NO 0 ALLOWED
800 002272 103407          BCS    100$
801
802 002274 111200          MOVB   (R2),R0    ;BUILD THE OFFSET
803 002276 160001          SUB    R0,R1     ;R1 IS 0
804 002300 060001          99$:  ADD    R0,R1
805 002302 077502          SOB   R5,99$
806 002304 010167 177062  MOV    R1,WHICH  ;LOAD IT
807 002310 000241          CLC
808 002312 000207          100$: RTS        PC
809
810          ;INPUT AN OCTAL BYTE
811 002314 004767 000054  ITYP4: JSR    PC,OCTIN ;READ THE NUMBER
812 002320 103404          BCS    100$      ;RETURN IF FAILURE
813 002322 022705 000377  CMP    #377,R5   ;TOO LARGE?
814 002326 103401          BCS    100$      ;YES, RETURN FAILURE
815 002330 110513          MOVB   R5,(R3)   ;LOAD IT IN MEMORY
816 002332 000207          100$: RTS        PC

```

INITIALIZATION CODE

```

817
818
819 002334 004767 000034      ;INPUT AN OCTAL WORD
ITYP6: JSR      PC,OCTIN
820 002340 103403
      BCS      100$
821 002342 110523
      MOVB     R5,(R3)+      ;SAVE LOW BYTE
822 002344 000305
      SWAB     R5
823 002346 110513
      MOVB     R5,(R3)+      ;SAVE HIGH BYTE
824 002350 000207
100$:  RTS      PC
825
826      ;ASCII CHARACTERS
ITYP10: CMPB     (R2),R0      ;SET C BIT IF GREATER
827 002352 121200
      BCS      100$
828 002354 103406
      MOV      #INSTR,R1      ;INPUT STRING
829 002356 012701 003350'
      MOVB     (R2)+,R0      ;COPY THIS MANY
830 002362 112200
      MOVB     (R1)+,(R3)+    ;COPY INPUT
831 002364 112123
      BEQ      100$          ;QUIT ON NULL
832 002366 001401
      SOB      R0,10$
833 002370 077003
      SOB      R0,10$
834 002372 000207
100$:  RTS      PC
835
836 002374 005005
OCTIN: CLR      R5          ;USE R5 FOR RESULT
837 002376 012700 003350'
      MOV      #INSTR,R0      ;POINT TO INPUT STRING
838 002402 112001
      MOVB     (R0)+,R1      ;GET NEXT CHARACTER
839 002404 001416
      BEQ      100$          ;IF 0, DONE
840 002406 162701 000060
      SUB      #'0,R1        ;SET C BIT IF LESS THAN '0
841 002412 103413
      BCS      100$          ;RETURN FAILURE
842 002414 022701 000007
      CMP      #7,R1         ;COMPARE WILL SET C BIT IF DIGIT > 7
843 002420 103410
      BCS      100$          ;RETURN FAILURE
844
845 002422 006305
      ASL      R5          ;MULTIPLY BY 8 AND ADD IN NEW DIGIT
846 002424 103406
      BCS      100$          ;IF MORE THAN 16 BITS ARE ENTERED, RETURN C SET
847 002426 006305
      ASL      R5
848 002430 103404
      BCS      100$
849 002432 006305
      ASL      R5
850 002434 103402
      BCS      100$
851 002436 050105
      BIS      R1,R5        ;ADD CURRENT DIGIT
852 002440 000760
      BR       10$          ;DO NEXT
853
854 002442 000207
100$:  RTS      PC          ;DONE, C BIT HOLDS SUCCESS CODE

```

INITIALIZATION CODE

```

856 ; MAIN TERMINAL I/O ROUTINES
857
858 ;PUT A COMMAND STRING INTO INBUF, RETURN C SET IF NO CHARACTERS ENTERED
859 ;USES R0, R1, R2
860 ;RETURNS NUMBER OF CHARACTERS ENTERED, NOT INCLUDING TRAILING NULL, IN R0
861 ;(AND Z SET IF NO CHARS ARE ENTERED)
862 002444 012700 012442' RETCOM: MOV #RETMMSG,R0 ;FREQUENTLY USED MESSAGE
863 002450 GETCOM: .TYPMSG ;OUTPUT A MESSAGE
864 002454 012702 000020 MOV #20,R2 ;BUFFER LENGTH
865 002460 012700 003350' MOV #INSTR,R0
866 002464 005010 CLR (R0) ;MAKE SURE WHOLE FIRST WORD IS NULL
867 002466 105010 10$: CLRB (R0) ;NULL TRAILER
868 002470 004767 000200 JSR PC,XCHK ;GET CHAR IN R1, HANDLE XON/XOFF, +C
869 002474 005701 TST R1
870 002476 002773 BLT 10$
871
872 002500 120127 000177 CMPB R1,#DEL ;IS CHARACTER A DELETE?
873 002504 001016 BNE 20$ ;IF NOT, PROCEED
874 002506 022700 003350' CMP #INSTR,R0 ;ANY CHARACTERS IN BUFFER?
875 002512 001765 BEQ 10$ ;NO, GET THE NEXT CHAR
876 002514 105704 TSTB R4 ;IS THIS THE FIRST DELETE?
877 002516 100402 BMI 15$ ;NO, DON'T PRINT SLASH
878 002520 004767 000130 JSR PC,SLASH
879 002524 114001 15$: MOVB -(R0),R1 ;REECHO DELETED CHARACTER, DEC POINTER
880 002526 004767 000126 JSR PC,TTY
881 002532 052704 000200 BIS #BIT7,R4 ;MARK DELETE STATE
882 002536 005202 INC R2 ;FIX BUFFER LENGTH POINTER
883 002540 000752 BR 10$
884
885 002542 105704 20$: TSTB R4 ;WAS PREVIOUS CHAR A DELETE?
886 002544 100006 BPL 30$ ;IF NO, BRANCH
887 002546 042704 000200 BIC #BIT7,R4 ;SIGNAL FINISHED
888 002552 010146 MOV R1,-(SP) ;END OF DELETE
889 002554 004767 000074 JSR PC,SLASH
890 002560 012601 MOV (SP)+,R1
891 002562 120127 000101 30$: CMPB R1,#'A
892 002566 002402 BLT 40$
893 002570 042701 000040 BIC #BIT5,R1 ;FORCE CAPITALIZATION
894 002574 004767 000060 40$: JSR PC,TTY ;ECHO CHARACTER
895
896 002600 120127 000012 CMPB R1,#L ;LINE FEED - IGNORE
897 002604 001730 BEQ 10$
898
899 002606 020127 000015 CMP R1,#C ;CARRIAGE RETURN
900 002612 001405 BEQ 99$
901
902 002614 110120 MOVB R1,(R0)+ ;STORE CHARACTER
903 002616 077255 SOB R2,10$ ;OTHERWISE, GET NEXT
904 002620 012700 012642' MOV #TOOLNG,R0
905 002624 000711 BR GETCOM
906
907 002626 162700 003350' 99$: SUB #INSTR,R0 ;RETURN NUMBER OF CHARACTERS
908 002632 000207 RTS PC
909
910 ;PUT A MESSAGE IN R0 OUT TO TERMINAL, USES R0 AND R1
911 002634 004767 000034 TYPMSG: JSR PC,XCHK
912 002640 112001 MOVB (R0)+,R1

```


INITIALIZATION CODE

913	002642	001403		BEQ	10\$
914	002644	004767	000010	JSR	PC,TTY
915	002650	000771		BR	TYPMSG
916	002652	000207		RTS	PC

10\$:

INITIALIZATION CODE

```

918 ;SUPPORT I/O FUNCTIONS
919
920 ;OUTPUT A SLASH - IT IS UP TO THE CALLER TO SAVE R1
921 002654 112701 000057 SLASH: MOVB #'/',R1 ;SET UP FOR AND FALL INTO TTY ROUTINE
922
923 ;OUTPUT THE SINGLE CHARACTER IN R1
924 002660 105737 177564 TTY: TSTB @#OUTSTA
925 002664 100375 BPL TTY
926 002666 110137 177566 MOVB R1,@#OUTBUF
927 002672 000207 RTS PC
928
929 ;CHECK FOR XON OR XOFF
930 002674 012701 177777 XCHK: MOV #-1,R1 ;NO CHARACTER
931 002700 105737 177560 TSTB @#INSTA ;HAS ANYTHING NEW COME IN?
932 002704 100023 BPL 99$
933 002706 113701 177562 MOVB @#INBUF,R1 ;GET NEW CHARACTER
934 002712 120127 000023 CMPB R1,#XOFF
935 002716 001003 BNE 10$
936 002720 052704 100000 BIS #BIT15,R4 ;MARK XOFF
937 002724 000763 BR XCHK ;GET NEXT CHAR
938 002726 120127 000021 10$: CMPB R1,#XON
939 002732 001003 BNE 20$
940 002734 042704 100000 BIC #BIT15,R4 ;CLEAR XOFF BIT
941 002740 000755 BR XCHK
942
943 002742 120127 000003 20$: CMPB R1,#CTC ;CHECK FOR CONTROL C
944 002746 001002 BNE 99$
945 002750 000167 000200 JMP 200
946
947 002754 005704 99$: TST R4 ;CHECK FOR CURRENT XOFF
948 002756 100746 BMI XCHK ;YES - IGNORE THIS CHARACTER AND GET NEXT
949 002760 000207 RTS PC

```

INITIALIZATION CODE

```

951          ; OTHER UTILITIES
952
953 002762   EXIT:      ;EXIT IS THE SAME THING AS POSITIVE COMPLETION
954 002762   YUP:      CMP      (SP)+,(SP)+      ;RETURN YES FROM MENU
955 002764   CLC
956 002766   RTS      PC
957
958 002770   NOPE:     CMP      (SP)+,(SP)+      ;RETURN NO FROM MENU
959 002772   SEC
960 002774   RTS      PC
961
962          ;MAKE SURE THE FILE ENTERED IS CORRECT
963          ;FILE MUST BE ENTERED IN THE FORM DDN:ABCDEF.GHI, WHERE DDN: IS THE
964          ; DEVICE NAME OF A SUPPORTED DEVICE, ABCDEF IS A STRING OF 6 OR FEWER
965          ; ALPHANUMERIC CHARACTERS, AND .GHI IS AN OPTIONAL FILE TYPE. IF .GHI
966          ; IS OMITTED IT WILL DEFAULT TO .SCF
967
968          ;FILE NAME IS ALREADY IN INSTR, RETURNS C CLEAR IF GOOD, SET IF NO GOOD
969          ;USES REGISTERS R0, R1, R2, R3
970
971          ;VER2CHKFIL:  MOV      #INSTR,R0      ;INPUT STRING
972          ;VER2      JSR      PC,SETDDB        ;RETURN R5 POINTING TO GOOD DDB
973          ;VER2      BCS      100$
974          ;VER2
975          ;VER2      MOV      #DDB+XXNAM,R2     ;COPY FILE NAME INTO DDB
976          ;VER2      MOV      R0,R3           ;FORMAT FOR OUTPUT TOO
977          ;VER2      MOV      #6,R1           ;MAXIMUM OF 6 CHARACTERS IN NAME
978          ;VER2      JSR      PC,FIL1
979          ;VER2      TSTB     (R0)
980          ;VER2      BNE      30$
981          ;VER2      MOV      #SCF,R0         ;IF NO EXTENSION, DEFAULT TO .SCF
982          ;VER2      BR       40$
983          ;VER2
984          ;VER230$:    CMPB     (R0),#' '      ;LAST CHAR MUST BE NULL OR .
985          ;VER2      BNE      99$            ;NO? FAIL.
986          ;VER240$:    MOV      #4,R1         ;MAX OF 4 CHARS IN EXTENSION
987          ;VER2      JSR      PC,FIL2
988          ;VER2      CLRB     (R3)           ;NULL TRAIL JUST TO BE SAFE
989          ;VER2      TSTB     (R0)           ;FINAL CHAR MUST BE NULL
990          ;VER2      BEQ      100$
991          ;VER299$:    SEC
992          ;VER2100$:   RTS      PC
993
994          ;THIS IS CALLED TO CHECK THE FILE NAME. IT TAKES IN R1 AND R0 A POINTER TO THE
995          ;FILE NAME, IN R1 THE LENGTH OF THE CURRENT FIELD, AND IN R2 THE LOCATION TO
996          ;COPY THE FILE NAME TO. IT BLANK FILLS (R2) TO THE LENGTH OF R1 FROM THE
997          ;FIRST NON ALPHANUMERIC CHAR IN (R0), AND FORMATS INSTR CORRECTLY FOR
998          ;OUTPUT
999          ;VER2.ENABL  LSB
1000         ;VER2FIL1:  CMPB     (R0),#'Z
1001         ;VER2      BGT      100$          ;THIS MUST BE ILLEGAL CHAR, JUST RETURN
1002         ;VER2      CMPB     (R0),#'A
1003         ;VER2      BGE      FIL2          ;IF B, MUST BE ALPHABETIC
1004         ;VER2
1005         ;VER2      CMPB     (R0),#'9
1006         ;VER2      BGT      100$          ;IF NOT A LETTER MUST BE A NUMBER
1007         ;VER2      CMPB     (R0),#'0          ;MUST BE ILLEGAL

```

INITIALIZATION CODE

```

1008 ;VER2 BLT 20$ ;MIGHT BE NULL OR ' ', SO BLANK PAD (R2)
1009 ;VER2FIL2: MOVB (R0),(R2) ;ALPHANUMERIC - COPY
1010 ;VER2 MOVB (R0),(R3) ;COPY IT INTO INSTR TOO
1011 ;VER2 SOB R1,FIL1 ;NEXT CHAR
1012 ;VER2 BR 100$ ;DONE
1013 ;VER2
1014 ;VER220$: MOVB 0' ,(R2) ;FILL WITH BLANKS
1015 ;VER2 SOB R1,20$
1016 ;VER2100$: RTS PC
1017 ;VER2.DSABL LSB
1018
1019 ;R2 CONTAINS 2 LETTER DEVICE NAME, RETURN DEVICE DDB SET UP FOR DEVICE, AND
1020 ; R5 POINTING TO IT, R0 POINTING TO FRONT OF FILE NAME
1021 ;VER2SETDDB: MOV (R0),R2
1022 ;VER2 MOV @DEV,R1 ;LOAD POSSIBLE DEVICES
1023 ;VER210$: CMP (R1),R2 ;DO WORD COMPARE FOR 2 LETTER DEVICE
1024 ;VER2 BEQ 20$ ;MATCH
1025 ;VER2 CMP (R1),(R1) ;DONE?
1026 ;VER2 BLT 10$ ;NO - TRY NEXT DEVICE
1027 ;VER2 BR 99$
1028 ;VER2
1029 ;VER220$: MOVB (R0),R2 ;CHECK UNIT NUMBER AND COLON
1030 ;VER2 SUB 0'R2 ;LEAVE BINARY UNIT NUMBER IN R2
1031 ;VER2 BLT 99$
1032 ;VER2 CMP R2,07
1033 ;VER2 BGT 99$
1034 ;VER2 CMPB (R0),0'
1035 ;VER2 BNE 99$
1036 ;VER2
1037 ;VER2 MOV @DDB,XBUF,R3 ;GOOD DEVICE, BUILD DDB
1038 ;VER2 MOV @BUF,(R3)
1039 ;VER2 MOV @XXNAM-XBUF-2,R5 ;LOAD BUFFER SEPERATELY
1040 ;VER2 MOV (R1),R1 ;LOAD THE DDB FRONT POINTER
1041 ;VER230$: MOVB (R1),(R3)
1042 ;VER2 SOB R5,30$
1043 ;VER2 MOV @DDB,R5
1044 ;VER2 MOVB R2,XDN(R5) ;LOAD DRIVE NUMBER
1045 ;VER2 BR 100$
1046 ;VER2
1047 ;VER299$: SEC
1048 ;VER2100$: RTS PC
1049 ;VER2
1050 ;VER2DEV: .ASCII /DD/
1051 ;VER2 .WORD DYDDB-XBUF+2 ;DYDDB (SKIP THE BUFFER)
1052 ;VER2 .ASCII /DY/
1053 ;VER2 .WORD DYDDB-XBUF+2 ;DYDDB
1054 ;VER2 .ASCII /DX/
1055 ;VER2 .WORD DYDDB-XBUF+2 ;DYDDB
1056 ;VER2 .WORD 0
1057 ;VER2SCF: .ASCIIZ /.SCF/
1058 ;VER2 .EVEN
1059
1060 ;WRITE EAROM
1061 002776 WRTROM: JSR PC,BLDCHK ;BUILD, LOAD CHECKSUM
1062 002776 004767 000206 CLR R1 ;PAGE
1063 003002 005001 MOV @UPDATE,R2
1064 003004 012702 000000G

```

INITIALIZATION CODE

```

1065 003010 052737 000020 177520      BIS      #BIT4,#BCSR      ;WRITE ENABLE
1066 003016 032704 000004          BIT      #BIT2,R4        ;SIZE OF ROM
1067 003022 001423          BEQ      EIGHTK         ;IF SET, ROM IS 2K
1068
1069 003024 012700 165000      10$:    MOV      #165000,R0    ;FIRST ADDRESS OF ROM
1070 003030 110137 177522      MOVB     R1,#PCRL0
1071 003034 121210      20$:    CMPB     (R2),(R0)
1072 003036 001403          BEQ      30$
1073 003040 111210      MOVB     (R2),(R0)      ;COPY
1074 003042 004767 000126      JSR      PC,DELAY
1075 003046 005202      30$:    INC      R2
1076 003050 005720      TST      (R0).         ;ROM IS 8X8K - HIGH BYTE IS NONEXISTANT
1077 003052 030027 000777      BIT      R0,#777       ;CHECK FOR END OF PAGE
1078 003056 001366      BNE      20$           ;READ NEXT BYTE
1079 003060 005721      TST      (R1).         ;NEXT PAGE (INC BY 2)
1080 003062 020127 000020      CMP      R1,#<PAGES*2>/4 ;ONLY DO 2K
1081 003066 001356      BNE      10$           ;COPY NEXT PAGE
1082 003070 000437          BR       LEAVE
1083
1084 003072 005003      EIGHTK: CLR      R3      ;ANY CHANGES YET?
1085 003074 110137 177522      10$:    MOVB     R1,#PCRL0    ;SET ROM PAGE
1086 003100 012700 165000      MOV      #165000,R0    ;FIRST ADDRESS OF ROM
1087 003104 122220      20$:    CMPB     (R2).,(R0).
1088 003106 001401      BEQ      30$
1089 003110 005203      INC      R3            ;CHANGES
1090 003112 105720      30$:    TSTB     (R0).         ;ROM IS 8X8K - HIGH BYTE IS NONEXISTANT
1091 003114 032700 000037      BIT      #37,R0        ;WRITE 20 BYTE BLOCKS
1092 003120 001371      BNE      20$
1093 003122 005703      TST      R3            ;FINISHED COMPARING A BLOCK - CHANGES?
1094 003124 001412      BEQ      50$           ;NO, GO DO NEXT
1095 003126 012703 000020      MOV      #20,R3        ;YES - RESET ROM AND RAM POINTERS
1096 003132 162700 000040      SUB      #40,R0
1097 003136 160302      SUB      R3,R2
1098 003140 112022      40$:    MOVB     (R0).,(R2).   ;COPY QUICKLY
1099 003142 105720      TSTB     (R0).
1100 003144 077303      SOB      R3,40$
1101 003146 004767 000022      JSR      PC,DELAY      ;WAIT FOR WRITE TO FINISH
1102
1103 003152 030027 000777      50$:    BIT      R0,#777       ;REACHED END OF PAGE
1104 003156 001352      BNE      20$           ;READ NEXT BYTE
1105 003160 005721      TST      (R1).         ;NEXT PAGE (INC BY 2)
1106 003162 020127 000100      CMP      R1,#PAGES*2
1107 003166 001342      BNE      10$           ;COPY NEXT PAGE
1108
1109 003170 000167 177566      LEAVE:   JMP      YUP    ;EXIT WITH CC
1110
1111      ;DELAY ABOUT 10 MS
1112 003174 010046      DELAY:  MOV      R0,-(SP)
1113 003176 012700 027340      MOV      #12000.,R0
1114 003202 077001      SOB      R0.
1115 003204 012600      MOV      (SP).,R0
1116 003206 000207      RTS      PC
1117
1118 003210 012700 000000G      BLDCHK: MOV      #UPDATE,R0 ;BUILD THE CHECKSUM
1119 003214 005002      CLR      R2
1120 003216 012703 000150      MOV      #104.,R3      ;NUMBER OF BYTES TO USE
1121 003222 112001      10$:    MOVB     (R0).,R1

```

INITIALIZATION CODE

```

1122 003224 060102          ADD    R1,R2
1123 003226 077303          SOB    R3,10$
1124 003230 005402          NEG    R2
1125 003232 110267 000150G  MOVB  R2,UPDATE+104.
1126 003236 000207          RTS    PC
1127
1128                          ;READ EAROM
1129 003240          REAROM:
1130 003240 012737 002770' 000004  MOV    #NOPE,#4
1131 003246 012702 000000G  MOV    #IMAGE,R2
1132 003252 005001          CLR    R1
1133 003254 110137 177522 10$:  MOVB  R1,#PCRLO
1134 003260 012700 165000  MOV    #165000,R0
1135 003264 111062 000000C 20$:  MOVB  (R0),UPDATE-IMAGE(R2)
1136 003270 112022  MOVB  (R0)+,(R2)+
1137 003272 105720  TSTB  (R0)+
1138 003274 030027 000777  BIT    R0,#777
1139 003300 001371  BNE   20$
1140 003302 005721  TST   (R1)+
1141 003304 020127 000100  CMP   R1,#PAGES*2
1142 003310 001361  BNE   10$
1143
1144 003312 012702 000000G  MOV    #UPDATE,R2
1145 003316 012701 004000G  MOV    #UPDATE+4000,R1
1146 003322 012700 002000  MOV    #4000/2,R0
1147 003326 022122 30$:  CMP   (R1)+,(R2)+
1148 003330 001003  BNE   100$
1149 003332 077003  SOB   R0,30$
1150 003334 052704 000004  BIS   #BIT2,R4
1151 003340 000241 100$:  CLC
1152 003342 000207  RTS    PC
1153

```

```

;2'S COMP
;LOAD CHECKSUM
;RETURN RO

;SET UP ERROR RETURN
;ADDRESS OF ROM IMAGE IN MEMORY
;PAGE
;SET ROM PAGE
;FIRST ADDRESS OF ROM
;MAKE COPY TO UPDATE
;MAKE IMAGE COPY
;ROM IS 8X8K - HIGH BYTE IS NONEXISTANT
;REACHED END OF PAGE?
;READ NEXT BYTE
;NEXT PAGE (INC BY 2)

;COPY NEXT PAGE

;CHECK FOR 8K ROM

;ANY DIFFERENCE MEANS 8K

;CHECK 2K
;MARK AS 2K ROM
;GOOD RETURN
;

```

INITIALIZATION CODE

```

1155 003344 000000      INISTK: .WORD
1156 003346 000000      MONADR: .WORD
1157 003350              INSTR: .BLKB 20 ;INPUT STRING
1158                    ;DUMMY APT COMMUNICATION AREA, -1'S ARE TEST TIME INFORMATION, MADE AS LARGE AS
1159                    ;POSSIBLE
1160 003370 000000 000000 177777 APTINF: .WORD 0,0,-1,-1,-1,0
1161
1162                    ROMLEN = 1000
1163                    ROMADR = 165000
1164                    ;VER2 .BLKB -XBUF
1165                    ;VER2DDB: .BLKB XXNAM.10.
1166                    ;VER2 .BYTE 0

```

INITIALIZATION CODE

```

1168      .NLIST BEX
1169      ;MENUS
1170 003404      $MENU:
1171 003404      015      012      012      .ASCII <C><L><L><L><T><T><T>'MAIN COMMANDS MENU'<C><L><L>
1172 003440      104      117      040      .ASCII 'DO YOU WANT TO:'<C><L><L>
1173      ;VER2      .ASCII <T>'F ..... CREATE SYSTEM CONFIGURATION FILE FROM EAROM ?'<C><L><L>
1174      ;VER2      .ASCII <T>'C ..... COPY SYSTEM CONFIGURATION FILE TO EAROM ?'<C><L><L>
1175      ;VER2      .ASCII <T>'U ..... UPDATE SYSTEM MEMORIES OR PERIPHERALS DESCRIPTION ?'<C>
<L><L>
1176 003462      011      123      040      .ASCII <T>'S ..... UPDATE SYSTEM HARDWARE DESCRIPTION ?'<C><L><L>
1177 003553      011      104      040      .ASCII <T>'D ..... DISPLAY/PRINT SYSTEM DESCRIPTION ?'<C><L><L><L>
1178 003643      011      122      040      .ASCII <T>'R ..... DISPLAY THIS MENU ?'<C><L><L>
1179 003713      011      110      040      .ASCII <T>'H ..... HELP ?'<C><L><L>
1180 003746      011      105      040      .ASCIZ <T>'E OR <RETURN> .. EXIT ?'<C><L>
1181 004001      $HMENU:
1182 004001      015      012      012      .ASCII <C><L><L><L><L><T><T><T>'UPDATE HARDWARE MENU'<C><L><L>
1183 004040      104      117      040      .ASCII 'DO YOU WANT TO UPDATE:'<C><L><L>
1184 004071      011      103      011      .ASCII <T>'C CPU DESCRIPTION ?'<C><L><L>
1185 004120      011      125      011      .ASCII <T>'U UNIBUS DESCRIPTION ?'<C><L><L>
1186 004152      011      102      011      .ASCII <T>'B BOOT DEVICE DESCRIPTION ?'<C><L><L>
1187 004211      011      123      011      .ASCII <T>'S BOOT SWITCH SETTINGS ?'<C><L><L><L>
1188 004246      011      110      011      .ASCII <T>'H HELP ?'<C><L><L>
1189 004262      011      122      011      .ASCIZ <T>'R RETURN TO MAIN COMMANDS MENU ?'<C><L>
1190
1191 004326      $EMENU:
1192 004326      015      012      012      .ASCII <C><L><L><L><L><L>' EXIT'<L>
1193 004343      015      012      011      .ASCII <C><L>\ DO YOU WANT TO:\
1194 004365      015      012      011      .ASCII <C><L>\ R .... RETURN TO MAIN MENU\
1195 004423      015      012      011      .ASCII <C><L>\ W .... WRITE THE ROM FROM THE CURRENT DESCRIPTION AND EXIT\
1196 004521      015      012      011      .ASCIZ <C><L>\ E .... EXIT WITH NO UPDATE\<C><L>
1197
1198 004562      $HELP:
1199 004562      015      012      012      .ASCII <C><L><L><L><L><L><T><T><T>'HELP MENU'<C><L><L>
1200 004607      111      116      106      .ASCII 'INFORMATION AVAILABLE:'<C><L><L>
1201      ;VER2      .ASCII <T>'F ..... CREATING A SYSTEM CONFIGURATION FILE FROM EAROM. '<C><L><L>
1202      ;VER2      .ASCII <T>'C ..... COPYING A SYSTEM CONFIGURATION FILE TO EAROM. '<C><L><L>
1203      ;VER2      .ASCII <T>'U ..... UPDATING SYSTEM MEMORIES OR PERIPHERALS DESCRIPTION. '<C><L><L>
1204 004640      011      123      040      .ASCII <T>'S ..... UPDATING SYSTEM HARDWARE DESCRIPTION. '<C><L><L>
1205 004722      011      104      040      .ASCII <T>'D ..... DISPLAYING/PRINTING SYSTEM DESCRIPTION. '<C><L><L>
1206 005006      011      105      040      .ASCII <T>'E ..... EXIT. '<C><L><L>
1207 005030      011      110      111      .ASCIZ <T>'HIT <RETURN> TO RETURN TO MAIN MENU'<L><L>
1208      ;VER2 $MORP:      .ASCII <C><L>'DO YOU WANT TO UPDATE:'
1209      ;VER2      .ASCII <C><L>' M ..... MEMORY'
1210      ;VER2      .ASCIZ <C><L>' P ..... PERIPHERAL'
1211
1212      ;VER2 $SURE:      .ASCII <C><L>/ IS THIS WHAT YOU WANT?/
1213      ;VER2      .ASCII <C><L>/ Y ..... YES/
1214      ;VER2      .ASCII <C><L>/ N OR <RETURN> .. NO/<C><L>
1215      ;VER2      .ASCIZ //

```


INITIALIZATION CODE

```

1217
1218 005077 015 012 012 CPUTIT: ;DESCRIPTOR HEADER STRINGS CPU DESCRIPTION/<C><L><L><L>
1219 005130 000 .ASCII <C><L><L><L>/
1220 005131 015 012 174 CPUDES: .ASCIZ //
        .ASCII <C><L>/)IGNORE)POWER )REBOOT)MASTER)HALT )HALT )LINE )LINE )LINE )MAIN
)
1221 005242 015 012 174 .ASCII <C><L>/)BATT. )UP )MODE )GRANT )TRAP )ON )CLOCK )CLOCK )CLOCK )MEMOR
)
1222 005353 015 012 174 .ASCII <C><L>/)STATUS)MODE ) )BUS )OPTION)BREAK )STATUS)INTER.)CSR )TEST
)
1223 005464 015 012 000 .ASCIZ <C><L>/
1224 005467 015 012 012 UBATIT: .ASCII <C><L><L><L>/ UBA DESCRIPTION/<C><L><L><L>
1225 005520 000 .ASCIZ //
1226 005521 015 012 174 UBADES: .ASCII <C><L>/)UNIBUS) 18 )UNIBUS)/
1227 005551 015 012 174 .ASCII <C><L>/)CACHE)BIT )MEMORY)/
1228 005601 015 012 174 .ASCII <C><L>/) )MODE ) )/
1229 005631 015 012 000 .ASCIZ <C><L>/
1230 005634 015 012 012 BOTTIT: .ASCII <C><L><L><L>/ BOOT DEVICES DESCRIPTION/<C><L><L><L>
1231 005676 000 .ASCIZ //
1232 005677 015 012 174 BOTDES: .ASCII <C><L>/)ENTRY)DEVICE)PHYS.)LOG.) CSR )/
1233 005745 015 012 174 .ASCII <C><L>/)NO. ) )NO. )NO. )ADDRESS )/
1234 006013 015 012 000 .ASCIZ <C><L>/
1235
1236 006016 015 012 012 SWITIT: .ASCII <C><L><L><L>/ SWITCH SETTINGS /<C><L><L><L>
1237 006050 000 .ASCIZ //
1238 006051 015 012 174 SWIDES: .ASCII <C><L>/)SWI. )DEVICE)LOG.)/
1239 006077 015 012 174 .ASCII <C><L>/)NO. ) )NO. )/
1240 006125 015 012 000 .ASCIZ <C><L>/
1241

```

INITIALIZATION CODE

```

1243
1244 006130      015      012      125  UBAP0:  .ASCIZ  <C><L>\UNIBUS CACHE          (DISABLE/ENABLE)  [0-1] =\
1245 006211      015      012      061  UBAP1:  .ASCIZ  <C><L>\18 BIT MODE          (DISABLE/ENABLE)  [0-1] =\
1246 006272      015      012      125  UBAP2:  .ASCIZ  <C><L>\UNIBUS MEMORY TEST (DISABLE/ENABLE)  [0-1] =\
1247
1248 006353      015      012      111  CPUP0:  .ASCIZ  <C><L>\IGNORE BATTERY STATUS (DISABLE/ENABLE)  [0-1] =\
1249 006451      015      012      120  CPUP1:  .ASCIZ  <C><L>\POWER UP MODE          (DIALOGUE/TURNKEY/ODT/TRAP) [0-3] =\
1250 006547      015      012      122  CPUP2:  .ASCIZ  <C><L>\REBOOT MODE          (DIALOGUE/TURNKEY/ODT/TRAP) [0-3] =\
1251 006645      015      012      115  CPUP3:  .ASCIZ  <C><L>\MASTERSHIP GRANT COUNT (0/1/2/3/4/5/6/7) [0-7] =\
1252 006743      015      012      110  CPUP4:  .ASCIZ  <C><L>\HALT/TRAP OPTION        (HALT/TRAP) [0-1] =\
1253 007041      015      012      110  CPUP5:  .ASCIZ  <C><L>\HALT ON BREAK          (DISABLE/ENABLE)  [0-1] =\
1254 007137      015      012      114  CPUP6:  .ASCIZ  <C><L>\LINE CLOCK CSR          (ENABLE/DISABLE)  [0-1] =\
1255 007235      015      012      114  CPUP7:  .ASCIZ  <C><L>\LINE CLOCK INTERRUPT (DISABLE/ENABLE)  [0-1] =\
1256 007333      015      012      114  CPUP8:  .ASCIZ  <C><L>\LINE CLOCK SOURCE      (BEVENT/50HZ/60HZ/800HZ) [0-3] =\
1257 007431      015      012      115  CPUP9:  .ASCIZ  <C><L>\MAIN MEMORY TEST        (ENABLE/DISABLE)  [0-1] =\
1258
1259 007527      015      012      104  BOTP0:  .ASCIZ  <C><L>\DEVICE NUMBER TO CHANGE (1 TO 9) =\
1260 007601      015      012      124  BOTP1:  .ASCIZ  <C><L>\TWO LETTER DEVICE MNEMONIC =\
1261 007654      015      012      120  BOTP2:  .ASCIZ  <C><L>\PHYSICAL UNIT NUMBER    (0 TO 377) =\
1262 007727      015      012      114  BOTP3:  .ASCIZ  <C><L>\LOGICAL UNIT NUMBER     (0 TO 377) =\
1263 010002      015      012      103  BOTP4:  .ASCIZ  <C><L>\CSR ADDRESS             (0 TO 177777) =\
1264
1265 010055      015      012      123  SWIP0:  .ASCIZ  <C><L>\SWITCH SETTING TO CHANGE (1 TO 6) =\

```

INITIALIZATION CODE

```

1267                                     ;HELP MESSAGE STRINGS
1268
1269 010130      015      012      040 $MSG1: .ASCII <C><L>' F'
1270 010136      015      012      011      .ASCII <C><L><T>'USE TO : '
1271 010151      015      012      012      .ASCII <C><L><L><T>' - UPDATES THE SYSTEM CONFIGURATION EAROM WITH THE '
1272 010242      015      012      011      .ASCII <C><L><T>' CONTENTS OF A SYSTEM CONFIGURATION FILE (.SCF) .'
1273 010332      015      012      012      .ASCIZ <C><L><L><T>'FILE NAME FORMAT: DEVICE:FILENAME'<C><L>
1274 010403      015      012      040 $MSG2: .ASCII <C><L>' C'
1275 010411      015      012      011      .ASCII <C><L><T>'USE TO : '
1276 010424      015      012      012      .ASCII <C><L><L><T>' - CREATES A SYSTEM CONFIGURATION FILE (.SCF) WITH THE '
1277 010520      015      012      011      .ASCII <C><L><T>' CONTENTS OF THE SYSTEM CONFIGURATION EAROM.'
1278 010603      015      012      012      .ASCIZ <C><L><L><T>'FILE NAME FORMAT: DEVICE:FILENAME'<C><L>
1279 010654      015      012      040 $MSG3: .ASCII <C><L>' U'
1280 010662      015      012      011      .ASCII <C><L><T>'USE TO : '
1281 010675      015      012      012      .ASCII <C><L><L><T>' - ADD MEMORY/PERIPHERAL DESCRIPTOR TO EAROM.'
1282 010760      015      012      011      .ASCII <C><L><T>' - DELETE MEMORY/PERIPHERAL DESCRIPTOR FROM EAROM.'
1283 011047      015      012      011      .ASCII <C><L><T>' - MODIFY MEMORY/PERIPHERAL DESCRIPTOR ON EAROM.'
1284 011134      015      012      011      .ASCII <C><L><T>' - DISPLAY MEMORY/PERIPHERAL DESCRIPTORS ON EAROM.'
1285 011223      015      012      012      .ASCIZ <C><L><L><T>'FOR DETAILED INFORMATION REFER TO UTILITY USERS GUIDE.'<C><L>
1286 011320      015      012      040 $MSG4: .ASCII <C><L>' S'
1287 011326      015      012      011      .ASCII <C><L><T>'USE TO : '
1288 011341      015      012      012      .ASCII <C><L><L><T>'C - MODIFY CPU DESCRIPTOR ON EAROM.'
1289 011412      015      012      011      .ASCII <C><L><T>'U - MODIFY UNIBUS DESCRIPTOR ON EAROM.'
1290 011465      015      012      011      .ASCII <C><L><T>'B - MODIFY BOOT DEVICES ON EAROM.'
1291 011533      015      012      011      .ASCII <C><L><T>'S - MODIFY BOOT SWITCH MEANINGS ON EAROM.'
1292 011611      015      012      012      .ASCIZ <C><L><L><T>'FOR DETAILED INFORMATION REFER TO UTILITY USERS GUIDE.'<C><L>
1293 011706      015      012      040 $MSG5: .ASCII <C><L>' D'
1294 011714      015      012      011      .ASCII <C><L><T>'USE TO : '
1295 011727      015      012      012      .ASCIZ <C><L><L><T>' - DISPLAY THE SYSTEM DESCRIPTION ON THE CONSOLE/LINEPRINTER.
'<C><L>
1296 012035      015      012      040 $MSG6: .ASCII <C><L>' E'
1297 012043      015      012      011      .ASCII <C><L><T>'USE TO : '
1298 012056      015      012      011      .ASCII <C><L><T>' - SAVE NEW SELECTIONS ON EAROM AND RETURN TO XXDP+ MONITOR.'
1299 012157      015      012      011      .ASCII <C><L><T>' - RETURN TO XXDP+ MONITOR WITHOUT SAVING NEW SELECTIONS.'<C><L>
1300      .EVEN
1301

```

INITIALIZATION CODE

```

1303                                     ;OTHER ASCII STRINGS
1304 012260      015      012      113 $HELLO: .ASCIZ <C><L>'KDJ11-B EAROM MAINTENANCE UTILITY V1.0'<C><L>
1305 012333      015      012      104 $TYASK: .ASCIZ <C><L>'DO YOU WANT A HARDCOPY LISTING ( N/Y CR=Y ) ? '
1306 012414      015      012      116 ILLCOM: .ASCII <C><L>/NOT A VALID CHOICE, /
1307 012442      015      012      124 RETMSG: .ASCIZ <C><L>/TYPE <RETURN> TO RETURN TO MENU /
1308 012505      015      012      111 BADFIL: .ASCIZ <C><L>/ILLEGAL FILE DESCRIPTION/
1309 012540      015      012      012 GETFIL: .ASCIZ <C><L><L><L>/ENTER DEVICE AND FILE NAME DDN:XXXXXX OR <RETURN> TO CANCEL: /
1310 012642      015      012      103 TOOLNG: .ASCIZ <C><L>/COMMAND TOO LONG, PLEASE REENTER: /
1311 012707      015      012      105 PROMPT: .ASCIZ <C><L>/ENTER SINGLE CHARACTER COMMAND FOLLOWED BY <RETURN>: /
1312 012777      015      012      105 NOROM: .ASCIZ <C><L>/EAROM NOT PRESENT IN SYSTEM/<C><L>
1313 013037      015      012      111 ERRIN: .ASCIZ <C><L>/INCORRECT INPUT/
1314 013061      015      012      106 NOTYET: .ASCIZ <C><L>/FUNCTION NOT YET IMPLEMENTED/
1315 013120      015      012      007 RESTOR: .ASCIZ <C><L><BELL>/WRITE FAILURE, RESTORING ORIGINAL ROM/
1316 013171      174      040      LBL5: .ASCII /) /
1317 013173      040      040      000 BL4: .ASCIZ / /
1318 013176      174      VCRLF: .ASCII /)/
1319 013177      015      012      000 CRLF: .ASCIZ <C><L>//
1320 013202      015      012      012 READ: .ASCIZ <C><L><L>/READ /
1321 013213      015      012      012 WRITE: .ASCIZ <C><L><L>/WRITE /
1322                                     .END
000001

```

SYMBOL TABLE

ALC	=	177770	CPUP7	007235R	ITYP6	002334R	RESTOR	013120R	XER	=	177777			
APTINF		003370R	CPUP8	007333R	L	=	000012	RETCOM	002444R	XFLCNT	=	177722		
ARGS	=	000000	CPUP9	007431R	LBL5		013171R	RETMMSG	012442R	XLSTBK	=	177750		
BADFIL		012505R	CPUTIT	005077R	LEAVE		003170R	ROMADR	=	165000	XMFID	=	000026	
BCSR	=	177520	CRLF	013177R	MAINM		000410R	ROMLEN	=	001000	XNB	=	000022	
BEGIN		000000R	CTC	=	000003	MENU		000266R	RWE		000554R	XOFF	=	000023
BELL	=	000007	DDISP	=	177570	MH.CHR		000512R	SLASH		002654R	XON	=	000021
BIT0	=	000001	DECTAB		001760R	MONADR		003346R	SRH	=	177766	XRD	=	000012
BIT1	=	000002	DEL	=	000177	MSTAB		000700R	SVC	=	177772	XSV	=	177772
BIT10	=	002000	DELAY		003174R	MS.B		000666R	SWIDES		006051R	XSVBLK	=	177730
BIT12	=	010000	DESCRP		001374R	MS.C		000666R	SWIDT		001212R	XSVCNT	=	177726
BIT15	=	100000	DLT	=	177760	MS.CHR		000464R	SWIPO		010055R	XSVDAT	=	177740
BIT2	=	000004	DSWR	=	177570	MS.H		000366R	SWITIT		006016R	XSVEXT	=	177736
BIT3	=	000010	EIGHTK		003072R	MS.S		000666R	T	=	000011	XSVMAP	=	177724
BIT4	=	000020	EMENU		000536R	MS.U		000666R	TOOLNG		012642R	XSVNAM	=	177732
BIT5	=	000040	ERRIN		013037R	M.CHRS		000432R	TTY		002660R	XSVUPT	=	177752
BIT6	=	000100	ERROR		000346R	M.D		000560R	TYPMSG		002634R	XSVXX	=	177742
BIT7	=	000200	ERROR1		000342R	M.E		000646R	U	=	000174	XWC	=	000002
BLDCHK		003210R	ETR	=	177764	M.H		000402R	UBADES		005521R	XWCTR	=	177716
BLKID	=	000200	EXIT		002762R	M.S		000360R	UBADT		001072R	XWILD	=	177720
BL4		013173R	GETCOM		002450R	NOPE		002770R	UBAPO		006130R	XWT	=	000014
BOT	=	177754	GETFIL		012540R	NOROM		012777R	UBAP1		006211R	XXNAM	=	000024
BOTDES		005677R	HARDM		000440R	NOTYET		013061R	UBAP2		006272R	X1ST	=	177754
BOTDT		001132R	HELPM		000474R	OCTIN		002374R	UBATIT		005467R	X1STBK	=	177744
BOTPO		007527R	HMSG1		000600R	OCTOUT		002016R	UPDATE	=	***** GX	YN		000532R
BOTP1		007601R	HMSG2		000604R	OTYP0		001570R	UPDRAM		001252R	YORN		000516R
BOTP2		007654R	HMSG3		000610R	OTYP10		002100R	VCRLF		013176R	YUP		002762R
BOTP3		007727R	HMSG4		000614R	OTYP2		001646R	WHICH		001372R	ZER	=	177756
BOTP4		010002R	HMSG5		000620R	OTYP4		001772R	WRITE		013213R	\$EMENU		004326R
BOTTIT		005634R	HMSG6		000624R	OTYP6		002002R	WRTRM		002776R	\$HELLO		012260R
BYTBLK	=	001000	ILLCOM		012414R	OUTBUF	=	177566	XBA	=	000004	\$HELP		004562R
C	=	000015	IMAGE	=	***** GX	OUTMAP		000712R	XBC	=	000016	\$HMENU		004001R
CLS	=	177762	INBUF	=	177562	OUTSTA	=	177564	XBKLG	=	177746	\$MENU		003404R
CPUDES		005131R	INI	=	177774	OUT\$		000356R	XBT	=	177754	\$MSG1		010130R
CPUDT		000742R	INISTK		003344R	PAGES	=	000040	XBUF	=	177714	\$MSG2		010403R
CPUP0		006353R	INMAP		000724R	PAINT		000270R	XCHK		002674R	\$MSG3		010654R
CPUP1		006451R	INSTA	=	177560	PCALO	=	177522	XCM	=	000000	\$MSG4		011320R
CPUP2		006547R	INSTR		003350R	PROMPT		012707R	XCO	=	000010	\$MSG5		011706R
CPUP3		006645R	ITYP0		002144R	PRTCSR	=	000000	XDN	=	177776	\$MSG6		012035R
CPUP4		006743R	ITYP10		002352R	READ		013202R	XDR	=	000020	\$SWR	=	160000
CPUP5		007041R	ITYP2		002212R	REAROM		003240R	XDT	=	000006	\$TYASK		012333R
CPUP6		007137R	ITYP4		002314R	REPLOTT		000356R						

. ABS. 000000 000
 013225 001
 ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 9421 WORDS (37 PAGES)
 DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
 ELAPSED TIME: 00:00:24
 B4,B4/-SP=BLMAC,DDBDEF,B4,END

.NLIST

;MACRO-PACKAGE REVISION HISTORY

```

-----
;REVISION      DATE      CHANGE
-----
; 1.0          25-MAY-78   INITIAL ISSUE
; 1.1          31-MAY-78   DELETE $BKREAD AND $NXTBLK MACRO DEFINITIONS
; 1.2          6-JUN-78   ADDED $CHKSUM MACRO
; 1.3          8-JUL-78   DELETE .PTRRD, .BMOVE, .SCHAIN, AND .ABORT MACRO DEFINITIONS
;              8-JUL-78   ADD .SETERR AND .ERROR MACRO DEFINITIONS
; 2.0          8-JUL-78   FIELD TEST VERSION
; 2.1          12-JUL-78  ADD $ABORT DEFINITION; REMOVE $NXTBLK DEFINITION
; 2.2          22-AUG-78  ADD MACRO DEFINITIONS FOR ROUTINES MOVED FROM DRIVERS TO DRVCOM
; 3.0          20-JUL-80  MAKE CHANGES MARKED BY ;MAC001
;              28-OCT-80  REMOVED $REMAIN MACRO                      ;MAC002
;              31-OCT-80  ADDED .FRCTYP AND .CHKCC                   ;RWB001
;              31-OCT-80  CHANGED .ERROR TO USE R0                 ;RWB001
;              removed old EMT macros for blast program ;rwy
-----

```

;fake the needed EMT calls

```

;this macro recovers from errors during I/O
;this is supposed to pass errors my way using the calls already existing
; in the drivers

```

```

.macro .frctyp arg
    mov    arg,r0
    sec

```

.endm

```

.macro .upknam
    jsr    pc,upknam

```

.endm

```

.macro .cmpnam
    jsr    pc,cmpnam

```

.endm

; THE FOLLOWING MACROS GENERATE TRAP CALLS

;

```
.MACRO $TYPRAD ARG1
.NARG NARGS
.IF NE,NARGS
.NTYPE NTYPE,ARG1
.IF NE,NTYPE
MOV ARG1,R0
.ENDC
.ENDC
.ENDM $TYPRAD TRAP 1
```

```
.MACRO $BCDCV ARG1
.NARG NARGS
.IF NE,NARGS
.NTYPE NTYPE,ARG1
.IF NE,NTYPE
MOV ARG1,R0
.ENDC
.ENDC
.ENDM $BCDCV TRAP 2
```

```
.MACRO $DATUPK ARG1
.NARG NARGS
.IF NE,NARGS
.NTYPE NTYPE,ARG1
.IF NE,NTYPE
MOV ARG1,R0
.ENDC
.ENDC
.ENDM $DATUPK TRAP 3
```

```
.MACRO $BYTFIL
TRAP 4
.ENDM $BYTFIL
```

```
.MACRO $CLRBMP
TRAP 5
.ENDM $CLRBMP
```

```
.MACRO $BKREAD
TRAP 6
.ENDM $BKREAD
```

```
.MACRO $RDDAT
TRAP 7
.ENDM $RDDAT
```

```
.MACRO $WRDAT
TRAP 10
.ENDM $WRDAT

.MACRO $BLKWRT
TRAP 11
.ENDM $BLKWRT

.MACRO $WRTLCT
TRAP 12
.ENDM $WRTLCT

.MACRO $CLRBUF
TRAP 13
.ENDM $CLRBUF

.MACRO $PAKNAM
TRAP 14
.ENDM $PAKNAM

.MACRO $TYPNAM
TRAP 15
.ENDM $TYPNAM

.MACRO $MBUFAD
TRAP 16
.ENDM $MBUFAD

.MACRO $CHKSUM
TRAP 17
.ENDM $CHKSUM

.MACRO $BMOVE
TRAP 20
.ENDM $BMOVE

.MACRO $ABORT ARG
IF NB ARG
.TYPMSG ARG
.ENDC
TRAP 21
.ENDM $ABORT

.MACRO $DTDEL
TRAP 22
.ENDM $DTDEL
```



```
.MACRO $BOOT
  TRAP 23
.ENDM $BOOT

.MACRO $CLRMAP
  TRAP 24
.ENDM $CLRMAP

.MACRO $ALOCBK
  TRAP 25
.ENDM $ALOCBK

.MACRO $ALLOC
  TRAP 26
.ENDM $ALLOC

.MACRO $READMP
  TRAP 27
.ENDM $READMP

.MACRO $CLSMAP
  TRAP 30
.ENDM $CLSMAP

.MACRO $WRTMAP
  TRAP 31
.ENDM $WRTMAP

.MACRO $STUFDS
  TRAP 33
.ENDM $STUFDS

.MACRO $CLOSE
  TRAP 34
.ENDM $CLOSE

.MACRO $DALSBK
  TRAP 35
.ENDM $DALSBK

.MACRO $DALLNK
  TRAP 36
.ENDM $DALLNK

.MACRO $DALCTG
  TRAP 37
.ENDM $DALCTG

.MACRO $STMAPS
  TRAP 40
.ENDM $STMAPS

.LIST
```

.olist
; **
; DDB OFFSETS
; **

; MJD001
; MJD001
; MJD001

```

XBUF      = -64    ; INDEX TO DDB MONITOR BUFFER
XWCTR     = -62    ; INDEX TO WRITE COUNTER
XWILD     = -60    ; INDEX TO FILE MODE INDICATOR
XFLCNT    = -56    ; INDEX TO FILE COUNT
XSVMAP    = -54    ;
XSVCNT    = -52    ;
XSVBLK    = -50    ;
XSVNAM    = -46    ; PHONY UFD BLOCK POINTERS
XSVEXT    = -42    ;
XSVDAT    = -40    ;
XSVXX     = -36    ;
X1STBK    = -34    ;
XBKLG     = -32    ;
XLSTBK    = -30    ;
XSVUPT    = -26    ;
X1ST      = -24    ; INDEX TO FIRST POINTER
XBT       = -24    ; INDEX TO BOOT ROUTINE
BOT       = -24    ; INDEX TO BOOT ROUTINE
ZER       = -22    ; INDEX TO ZERO ROUTINE
DLT       = -20    ; INDEX TO DELETE ROUTINE
CLS       = -16    ; INDEX TO CLOSE ROUTINE
ETR       = -14    ; INDEX TO ENTER ROUTINE
SRH       = -12    ; INDEX TO LOKUP ROUTINE
ALC       = -10    ; INDEX TO ALLOCATE ROUTINE
SVC       = -6     ; INDEX TO SERVICE ROUTINE (DRIVER)
XSV       = -6     ; INDEX TO SERVICE ROUTINE (DRIVER)
INI       = -4     ; INDEX TO INIT ROUTINE
XDN       = -2     ; DRIVE NUMBER INDEX
XER       = -1     ; RESULT STATUS
XCM       = 0      ; INDEX TO COMMAND REGISTER
XWC       = 2      ; INDEX TO WORD COUNT
XBA       = 4      ; INDEX TO BUS ADDRESS
XDT       = 6      ; INDEX TO BLOCK NUMBER
XCO       = 10     ; INDEX TO COMMAND
XRD       = 12     ; INDEX TO READ COMMAND
XWT       = 14     ; INDEX TO WRITE COMMAND
XBC       = 16     ; INDEX TO REQUESTED BLOCK COUNT
XDR       = 20     ; INDEX TO 1ST DIR BLOCK POINTER
XNB       = 22     ; INDEX TO LAST BLOCK # ALLOCATED
XXNAM     = 24     ; INDEX TO ASCII NAME IN DDB

xmfid     =        26
bytblk    =        512.

```

; MJD001
; MJD001
; MJD001
; MJD001
; MJD001
; MJD001
; MJD001

```

: **
: General purpose equates
:
:
T      = 11      ;tab character
BELL   = 07      ;bell character
L      = 12      ;line feed character
C      = 15      ;carriage return character
U      = 174     ;vertical dash character
CTC    = 3       ;control-c
XON    = 21      ;output on
XOFF   = 23      ;output off
del    = 177     ;rubout
BCSR   = 177520  ;Boot/Diagnostic Status register
PCRL0  = 177522  ;Page Control register
BLKID  = 000200  ;UFD block is memory/peripheral
DSWR   = 177570  ;Hardware switch register
DDISP  = 177570  ;Hardware display register
$SWR   = 160000  ;Flags for APT control

pages  = 32.     ;number of pages in EEROM

bit0   = 1
bit1   = 2
bit2   = 4
bit3   = 10
bit4   = 20
bit5   = 40
bit6   = 100
bit7   = 200
bit10  = 2000
bit12  = 10000
bit15  = 100000

prtcsr = 0       ;??? dummy printer address
insta  = 177560
inbuf  = insta+2
outsta = 177564
outbuf = outsta+2
.list

```