

IDENTIFICATION

-----

PRODUCT CODE: MAINDEC-11-DXQLA-I-D  
PRODUCT NAME: DEC/V11 MONITOR LIBRARY  
DATE: MARCH 1977  
MAINTAINER: DIAGNOSTIC GROUP

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

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

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

COPYRIGHT (C) 1974, 1975, 1976, 1977  
BY DIGITAL EQUIPMENT CORPORATION

**CONTENTS**

-----

**QABM DEC/X11 STANDARD MONITOR**

**QAED DEC/X11 SHORT MONITOR**



62 ;A FEW DEFINITIONS,  
 63  
 64 000000 R0 =R0 ;GENERAL PURPOSE REGISTER 0  
 65 000001 R1 =R1 ;GENERAL PURPOSE REGISTER 1  
 66 000002 R2 =R2 ;GENERAL PURPOSE REGISTER 2  
 67 000003 R3 =R3 ;GENERAL PURPOSE REGISTER 3  
 68 000004 R4 =R4 ;GENERAL PURPOSE REGISTER 4  
 69 000005 R5 =R5 ;GENERAL PURPOSE REGISTER 5  
 70 000006 R6 =R6 ;GENERAL PURPOSE REGISTER 6  
 71 000006 SP =R6 ;POINTER TO HARDWARE (SYSTEM DEFAULT) STACK  
 72 000007 PC =R7 ;PROGRAM COUNTER  
 73  
 74 177776 PS =177776 ;PROCESSOR STATUS REGISTER  
 75 177776 PSW =177776 ;PROCESSOR STATUS WORD  
 76 177570 HARDSR =177570 ;HARDWARE SWITCH REGISTER  
 77  
 78 177572 SSR0 =177572 ;MEMORY MANAGEMENT REGISTER 0  
 79 172516 SSR3 =172516 ;MEMORY MANAGEMENT REGISTER 3  
 80  
 81 170200 MAPREG =170200 ;BASE ADDRESS OF UNIBUS MAP BOX (11/70)

82 076600 MED =76600 ;MAINTENANCE EXAMINE AND DEPOSIT INSTRUCTION (11/60)  
 83 000101 RDSERV =101 ;LOG SERVICE REG. ADR == READ  
 84 000102 RDPBA =102 ;LOG PHYSICAL BUSS ADR, REG. ADR, == READ  
 85 000103 RDCUA =103 ;LOG CURRENT MICRO ADR, REG. ADR, == READ  
 86 000104 RDFGIN =104 ;LOG FLAG/INTERRUPT REG. ADR, == READ  
 87 000022 RDWHMI =022 ;GEN. WHO AM I REG. ADR, == READ  
 88 000222 WTHWMI =222 ;GEN. WHO AM I REG. ADR, == WRITE  
 89  
 90 177772 PIRGRG =177772 ;PROGRAMMED INTERRUPT REQUEST REGISTER  
 91  
 92 140000 UMODE =140000 ;BIT CONFIGURATION TO INDICATE USER MODE IN PSW  
 93 040000 SMODE =400000 ;BIT CONFIGURATION TO INDICATE SUPERVISOR MODE IN PSW  
 94  
 95 172300 KIPDR0 =172300 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 0  
 96 172302 KIPDR1 =172302 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 1  
 97 172304 KIPDR2 =172304 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 2  
 98 172306 KIPDR3 =172306 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 3  
 99 172316 KIPDR7 =172316 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 7  
 100 172340 KIPAR0 =172340 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 0  
 101 172342 KIPAR1 =172342 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 1  
 102 172344 KIPAR2 =172344 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 2  
 103 172346 KIPAR3 =172346 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 3  
 104 172356 KIPAR7 =172356 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 7  
 105  
 106 177600 UIPDR0 =177600 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 0  
 107 177602 UIPDR1 =177602 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 1  
 108 177604 UIPDR2 =177604 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 2  
 109 177616 UIPDR7 =177616 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 7  
 110 177640 UIPAR0 =177640 ;USER INSTRUCTION PAGE ADDRESS REGISTER 0  
 111 177642 UIPAR1 =177642 ;USER INSTRUCTION PAGE ADDRESS REGISTER 1  
 112 177644 UIPAR2 =177644 ;USER INSTRUCTION PAGE ADDRESS REGISTER 2  
 113 177656 UIPAR7 =177656 ;USER INSTRUCTION PAGE ADDRESS REGISTER 7

```

114      000004      PIRQ8  =IOT      ;MAKE THE PIRQ CALL A TRAP THROUGH THE IOT VECTOR
115      000000      OPEN   =0        ;USE OPEN TO DECLARE ALL VARIABLES AS 0 BEFORE STARTING
116      100000      BIT15  =100000 ;BIT 15 DEFINITION
117      040000      BIT14  =40000  ;BIT 14 DEFINITION
118      020000      BIT13  =20000  ;BIT 13 DEFINITION
119      010000      BIT12  =10000  ;BIT 12 DEFINITION
120      004000      BIT11  =4000   ;BIT 11 DEFINITION
121      002000      BIT10  =2000   ;BIT 10 DEFINITION
122      001000      BIT9   =1000   ;BIT 9 DEFINITION
123      000400      BIT8   =400    ;BIT 8 DEFINITION
124      000200      BIT7   =200    ;BIT 7 DEFINITION
125      000100      BIT6   =100    ;BIT 6 DEFINITION
126      000040      BIT5   =40     ;BIT 5 DEFINITION
127      000020      BIT4   =20     ;BIT 4 DEFINITION
128      000010      BIT3   =10     ;BIT 3 DEFINITION
129      000004      BIT2   =4      ;BIT 2 DEFINITION
130      000002      BIT1   =2      ;BIT 1 DEFINITION
131      000001      BIT0   =1      ;BIT 0 DEFINITION
132
133      000340      PRIY7  =340    ;BIT CONFIGURATION FOR PRIORITY LEVEL 7 IN PSM
134      000300      PRIY6  =300    ;BIT CONFIGURATION FOR PRIORITY LEVEL 6 IN PSM
135      000240      PRIY5  =240    ;BIT CONFIGURATION FOR PRIORITY LEVEL 5 IN PSM
136      000200      PRIY4  =200    ;BIT CONFIGURATION FOR PRIORITY LEVEL 4 IN PSM
137
138      005746      PUSH   =005746 ;TST  -(SP) ;;USED TO ADD A WORD TO THE STACK
139      024646      PUSH2  =024646 ;CMP  -(SP),-(SP) ;;USED TO PUT TWO WORDS ON THE STACK
140      005726      POPSP  =005726 ;TST  (SP)+ ;;USED TO DELETE A WORD FROM THE STACK
141      022626      POPSP2 =022626 ;CMP  (SP)+,(SP)+;USED TO DELETE TWO WORDS FROM THE STACK
142
143      000100      IE      =BIT6   ;DEFINITION OF INTERRUPT ENABLE BIT FOR ALL PDP-11 DEVICES
144      000040      KBUFL  =32     ;DEFINITION OF KEYBOARD BUFFER LENGTH (IN WORDS)
145      000620      IOQL   =400    ;DEFINITION OF INPUT/OUTPUT SERVICE QUEUE LENGTH(IN BYTES)
146      001130      TYPQL  =600    ;DEFINITION OF TYPING QUEUE LENGTH (IN BYTES)
147      031222      TYP LIM =TYPEQ+TYPQL ;DEFINITION OF LAST WORD IN TYPE QUEUE
148      030072      IOQLIM =IOQ+IOQL ;DEFINITION OF LAST WORD IN I/O QUEUE
    
```

```

149      ;THE FOLLOWING DEFINITIONS ARE THE INDEX VALUES USED BY MODE 6 AND 7
150      ;MONITOR INSTRUCTIONS WHEN REFERRING TO LOCATIONS IN THE HEADER OF A MODULE
151
152      000005      XFLAG  =5      ;REFERENCE USED TO KEEP TRACK OF WRITE BUFFER USAGE
153      000020      STAT   =16     ;POINTS TO THE LOW BYTE OF THE MODULE STATUS WORD
154      000021      STAT1  =17     ;POINTS TO THE HIGH BYTE OF THE MODULE STATUS WORD
155      000022      INIT   =18     ;POINTS TO THE MODULE'S STARTING ADDRESS
156      000024      SPOINT =20     ;POINTS TO THE MODULE'S STACK POINTER
157      000026      PSCNT  =22     ;REFERENCES THE LOCATION KEEPING TRACK OF THE NUMBER OF PASSES
158      000030      EPCNT  =24     ;REFERENCES THE LOCATION KEEPING TRACK OF THE NUMBER OF ERRORS
159      000032      SVR0   =26     ;POINTS TO WHERE THE MODULE'S REGISTER 0 CONTENTS ARE STORED
160      000034      SVR1   =28     ;POINTS TO WHERE THE MODULE'S REGISTER 1 CONTENTS ARE STORED
161      000036      SVR2   =30     ;POINTS TO WHERE THE MODULE'S REGISTER 2 CONTENTS ARE STORED
162      000040      SVR3   =32     ;POINTS TO WHERE THE MODULE'S REGISTER 3 CONTENTS ARE STORED
163      000042      SVR4   =34     ;POINTS TO WHERE THE MODULE'S REGISTER 4 CONTENTS ARE STORED
164      000044      SVR5   =36     ;POINTS TO WHERE THE MODULE'S REGISTER 5 CONTENTS ARE STORED
165      000046      SVR6   =38     ;POINTS TO WHERE THE MODULE'S REGISTER 6 CONTENTS ARE STORED
166      000050      CSRA   =40     ;REFERENCES THE BASE ADDRESS OF THE MODULE'S DEVICES(CONTROL AND
167      000052      ACSR   =42     ;REFERENCES THE CONTENTS OF THE CONTROL AND STATUS REGISTER
168      000052      SBADR  =42     ;POINTS TO THE ADDRESS OF WHAT THE DATA SHOULD HAVE BEEN
169      000054      ASTAT  =44     ;POINTS TO ANY STATUS REGISTER NOT THE SAME AS THE CSR
170      000054      WASADR =44     ;POINTS TO THE ADDRESS OF WHAT THE DATA WAS
171      000056      ASB    =46     ;POINTS TO THE ACTUAL EXPECTED DATA
172      000060      AWAS   =48     ;POINTS TO THE ACTUAL RECEIVED DATA
173      000062      RSTRT  =50     ;POINTS TO THE MODULE RESTART ADDRESS
174      000064      RBUFVA =52     ;REFERENCES THE VIRTUAL ADDRESS OF THE READ BUFFER
175      000066      RRUFP  =54     ;POINTS THE LOCATION FOR THE PHYSICAL ADDRESS OF THE READ BUFFER
176      000070      RBUFE  =56     ;POINTS THE EXTENDED ADDRESS BITS OF THE READ BUFFER
177      000072      PRUFSZ =58     ;POINTS THE SIZE OF THE READ BUFFER
178      000074      WBUFPA =60     ;POINTS TO THE WRITE BUFFER PHYSICAL ADDRESS
179      000076      WBUFE  =62     ;POINTS TO THE EXTENDED ADDRESS BITS OF THE WRITE BUFFER
180      000100      WBUFRQ =64     ;REFERENCES THE AMOUNT OF WRITE BUFFER REQUESTED BY MODULE
181      000102      WBUFSZ =66     ;REFERENCES THE SIZE OF THE WRITE BUFFER
182
183
    
```

```

184 ;LOAD TRAP AND VECTOR AREA
185
186 ;=0
187 000000 000000 000000 .WORD 0,0
188 000004 015246 BUSEV; BUSERR ;BUS ERROR TRAP POINTER,
189 000006 000340 PRTY7
190 000010 015276 RESIV; RESINT ;RESERVED INSTRUCTION TRAP,
191 000012 000340 PRTY7
192 000014 005700 TRCV; TRCI ;TRACE TRAP POINTER,
193 000016 000000 0
194 000020 006416 IOTV; PIRQ, ;SERVICE PIRQ CALL
195 000022 000340 PRTY7 ;PREVENT LOWER LEVEL MODULE FROM COMING IN ON
196 ;TOP OF HIGHER LEVEL ONES
197 000024 015460 PWRV; PWRDN ;POWER FAIL POINTER,
198 000026 000340 PRTY7
199 000030 000032 EMTV; +2 ;EMT POINTER
200 000032 000000 HALT
201 000034 015134 TRPV; TRPINT ;TRAP POINTER,
202 000036 000000 0
203 000040 000040 .=40
204 000040 000000 .WORD OPEN ;LOAD MEDIUM INDICATOR,
205 000042 000000 DDPPTRI; .WORD 0 ;CHAIN MODE ONLY, POINTS TO DDPDM,
206 000044 000000 .WORD 0
207 000046 016134 LOGIC
208 000050 000000 OPEN
209 000052 000000 OPEN
210 000054 000000G CAST ;GLOBAL POINTER FOR CAST MODIFICATION
211 000056 000001 1 ;ALSO FOR CAST
212 000060 000060 .=60
213 000060 006552 KYBDV; KRSRVC ;KYRD INTERRUPT POINTER,
214 000062 000340 PRTY7
215 000064 001526 TTPIV; TTYSRV ;TELETYPE PRINTER SERVICE
216 000066 000200 PRTY4
    
```

```

217 ;FROM 70 THROUGH 776 FILLED WITH +2 AND HALT.
218
219
220
221 ;=114
222 000114 015672 PARVCT; PARERR ;PARITY ERROR INTERRUPT POINTER
223 000116 000340 PRTY7
224
225
226 ;LOAD STARTING ADDRESSES
227
228 ;=200
229 000200 000167 001330 JMP START ;GO TO START OF MONITOR,
230 001000 .=1000
231 001000 000167 000530 JMP START ;GO TO START OF MONITOR,
232
233
234 001004 000000 FAKESR; OPEN ;LOCATION OF PSEUDO SWITCH REG, WHEN USED
    
```

```

235 ;VARIABLE DATA AREA CLEARED EVERY TIME RUN IS ISSUED OR 'C' IS TYPED
236
237 001006 000 IOQUE; .BYTE OPEN ;IOQUE AND TYPQUE MUST BE IN SAME WORD!!
238 001007 000 TYPQUE; .BYTE OPEN
239 001010 000 SPCFLG; .BYTE OPEN ;SPCFLG AND DIRIND MUST BE IN SAME WORD!!
240 001011 000 DIRIND; .BYTE OPEN
241 001012 000 BKQUE; .BYTE OPEN
242 001013 000 BRAKE; .BYTE OPEN
243 001014 000 TTYBSY; .BYTE OPEN ;TELETYPE BUSY FLAG, 0= NOT BUSY.
244 001015 000 MODCNT; .BYTE OPEN
245 001016 000 MODCTR; .BYTE OPEN
246 001017 000 ERRIND; .BYTE OPEN ;0=ERROR, NOT 0=DATA ERROR.
247 001020 000 FILCTR; .BYTE OPEN
248 001021 000 MDXCTR; .BYTE OPEN ;EXTENDED I/O MODULES NOT YET RUN IN CURRENT BANK
249 001022 000 MDXCNT; .BYTE OPEN ;EXTENDED I/O MODULE COUNT
250 001023 000 WBLG; .BYTE OPEN ;CURRENT VALUE OF XFLAG BEING CHECKED FOR
251 001024 000 STOP; .BYTE OPEN ;STOP MODULES TO RELOCATE PROGRAM
252 001025 000 BKCNT; .BYTE OPEN ;BACKGROUND MODULE COUNT
253 001026 000 XCNT; .BYTE OPEN ;RUNNABLE EXTENDED MODULE COUNT (TO CALCULATE ROTCT)
254 001027 000 QUECTR; .BYTE OPEN
255 001030 000 XFRFLG; .BYTE OPEN
256 001031 000 ROTCT; .BYTE OPEN ;VALUE USED FOR BUFFER ROTATION TIMEOUT (16 TIMES
257 ;XCNT OR 377, WHICHEVER IS SMALLER)
258 001032 000 PASTOT; .BYTE OPEN ;ALTERNATE RELOCATION COUNTER- TOTAL ENDPASS
259 ;CALLS IN CURRENT BANK
260 001033 000 PASREL; .BYTE OPEN
261 001034 000 BKTIME; .BYTE OPEN ;BKMOD INSTR CTR- LOADED FROM BKTIME
262 001035 000 ROT; .BYTE OPEN ;WBUF ROTATION RATE CTR- LOADED FROM ROTN
263 .EVEN
264 001036 000000 KBPTR; OPEN
265 001040 000000 MODPTR; OPEN ;MODULE POINTER
266 001042 000000 ADDRLO; OPEN
267 001044 000000 ADDRHI; OPEN
268 001046 000000 NUMBER; OPEN
269 001050 000000 NUMBPA; OPEN
270 001052 000000 NUMBEA; OPEN
271 001054 000000 DSTADR; OPEN
272 001056 000000 SRETRN; OPEN
273 001060 000000 IOBKID; OPEN
274 001062 000000 YES; OPEN
275 001064 000000 TABADR; OPEN
276 001066 000000 RSTAT; OPEN
277 001070 000000 TRCPC; OPEN
278 001072 000000 TRCPSW; OPEN
279 001074 000000 IOQ1; OPEN ;I/O QUEUE POINTERS.
280 001076 000000 IOQ2; OPEN
281 001100 000000 TYPQ1; OPEN ;LOAD TYPE QUEUE POINTER
282 001102 000000 TYPQ2; OPEN ;UNLOAD TYPE QUEUE POINTER
283 001104 000000 TYPADR; OPEN
284 001106 000000 TYPREI; OPEN
285 001110 000000 TYPCTR; OPEN ;USED TO MONITOR CONSOLE TELETYPE TO PREVENT HANGING
286 001112 000000 OASCEA; OPEN ;USED TO HOLD EA RITS FOR OCTAL TO
287 ;ASCII CONVERSION
288
289 001114 000000 MODWPA; OPEN
290 001116 000000 MODWEA; OPEN
    
```

```

291 001120 000000 MODW12; OPEN
292 001122 000000 MODRPA; OPEN
293 001124 000000 MODREA; OPEN
294 001126 000000 MODR12; OPEN
295 001130 000000 DERRCT; OPEN
296 001132 000000 SAVLO; OPEN
297 001134 000000 SAVHI; OPEN
298 001136 000000 MODADR; OPEN ;ADDRESS OF CURRENTLY EXECUTING (IOQ LEVEL) MODULE
299 001140 000000 TYPP1; OPEN ;SAVE LOCATIONS FOR TYPE ROUTINE REGISTERS
300 001142 000000 TYPR2; OPEN
301 001144 000000 TYPR3; OPEN
302 001146 000000 TYPR4; OPEN
303 001150 000000 TYPR5; OPEN
304 001152 000000 SPSAV; OPEN
305 001154 000000 ERRPOS; OPEN
306 001156 000000 CDFLG; OPEN ;INDICATE RETURN TO CHECKDATA ROUTINE
307 001160 000000 CCSRA; OPEN
308
309
    
```

```

;VARIABLE AND FIXED DATA AREA NOT CLEARED WHEN RUN OR *C IS ISSUED
310
311
312 001162 177560      TKS: 177560
313 001164 177562      TKB: 177562
314 001166 177564      TPS: 177564      ;IF THIS AND NEXT LOC ARE CHANGED,
315 001170 177566      TPB: 177566      ;CHANGE LOC,S LPWK1 AND LPWK2 IF PRESENT
316 001172 177514      LPS: 177514
317 001174 177516      LPB: 177516
318 001176 177570      SR: 177570
319 001200 123456      RANNUM: 123456      ;ADDRESS OF THE SWITCH REGISTER
320 001202 022333      RANWRK: 22333      ;RANDOM NUMBER IS CALC'D AND PASSED HERE
321 001204 000000      MENVA: OPEN        ;ALSO USED BY RAND.
322 001206 000000      MEMPA: OPEN
323 001210 000000      MEMEA: OPEN
324 001212 000000      MEM12: OPEN
325 001214 000000      MEMSBE: OPEN
326 001216 000000      MEMWAS: OPEN
327 001220 000000      LOCORE: OPEN      ;STARTING ADDR OF FREE CORE (MULTIPLE OF 1000 OCTAL)
328 001222 000000      ZERO: 0           ;LOCORE EA BITS
329 001224 000000      LCOR12: OPEN     ;TOP 12 BITS OF LOW CORE ADDRESS
330 001226 000000      HCOR12: OPEN     ;CONTAINS ST. ADDR OF HIGHEST BUFFER,
331                                     ;(TOP 12 BITS ONLY)
332 001230 000000      HCORSV: OPEN
333 001232 001534      WBUF: START       ;CONTAINS CURRENT WRITE BUFFER ADDR.
334 001234 000000      WBFEA: OPEN      ;WRITE BUFFER EA BITS (SHIFTED TO POS 4,5)
335 001236 000000      WBUF12: OPEN     ;TOP 12 BITS OF WBUF ADDRESS IF USING ROTATION
336 001240 000000      OFFMX: OPEN      ;TOP ALLOWABLE PROGRAM OFFSET
337 001242 000000      OFFSET: OPEN     ;CURRENT PROGRAM OFFSET
338 001244 000000      OFFSTD: OPEN     ;DESTINATION OFFSET
339 001246 000000      OFFSTM: OPEN     ;SAVE LOCATION FOR DESTINATION OFFSET
340                                     ;DURING MOVE
341 001250 000000      MOVBNK: OPEN     ;BANK BEING MOVED (SOURCE ADDRESS)
342 001252 000000      MAPRET: OPEN     ;RETURN ADDRESS FROM MPCODE ROUTINE
343 001254 000024      ERRLIM: 20.
344 001256 012000      MXWRF: AMODNM-START/2&177000 ;MAX SIZE OF WRITE BUFFER, VARIES
345                                     ;IF BUFFER ROTATION IS ENABLED
346 001260 000000      TTYBT: OPEN
347 001262 000000      CXRET: OPEN     ;CTRLX RETURN ADDRESS
348 001264 000000      SYSCNT: OPEN     ;TOTAL NUMBER OF SYSTEM ERRORS
349 001266 000000      PWRCNT: OPEN     ;TOTAL NUMBER OF POWER FAILS
350 001270 000000      BKPTI: OPEN     ;QUEUE POINTER INITIALIZATION FOR RUNNING BKMODS
351 001272 000000      KTSAV: 0        ;USED FOR SECOND PASS IN CHAIN MODE TO
352 001274 000000      HCR12A: 0       ;RESTORE TRUE STATE OF SYSTEM
353 001276 177600      PDRTAB: 177600
354 001300 172200
355 001302 172300      PDREND: 172300
356 001304 000000      KTPRES: OPEN     ;0= INHIBITED OR NOT PRESENT (LEAVE AS FULL
357                                     ;WORD FOR EASY MODIFICATION AND TO GUARANTEE
358                                     ;THAT FILCNT AND FILLFR ARE IN SAME WORD)
359 001306 014        FILCNT: .BYTE 12. ;FILCNT AND FILLFR MUST BE IN SAME WORD
360 001307 000        FILLER: .BYTE 0
361 001310 000        SYSERI: .BYTE OPEN
362 001311 000        RMODE: .BYTE OPEN ;RUN MODE INDICATOR
363 001312 000        CHN: .BYTE OPEN
364 001313 000        POTI: .BYTE OPEN
365 001314 000        ROTLI: .BYTE OPEN ;WRITE BUFFER ROTATION LOCKED FLAG
  
```

```

366 001315 000        RELOC: .BYTE OPEN ;RELOCATION ALLOWED FLAG
367 001316 000000      ROTCNT: OPEN     ;ROTATION COUNTER USED WITH ROTL
368 001320 000100      ROTNUM: 100      ;ROTATION CONSTANT USED WITH ROTCNT
369 001322 000400      ROTSIZ: 400     ;ROTATE PROGRAM BY *K
370 001324 000        FILLID: .BYTE OPEN ;FILL INDICATOR.
371 001325 000        MOVING: .BYTE OPEN ;CODE BEING RELOCATED
372 001326 000        RUNRFL: .BYTE OPEN
373 001327 000        LOCK: .BYTE OPEN ;PREVENT ROTATION OF CODE EXECUTION
374 001330 000        MVDOWN: .BYTE OPEN
375 001331 000        RETRY: .BYTE OPEN ;RETRY COUNT FOR CODE RELOCATION
376 001332 010        QUECNT: .BYTE 10 ;# OF TIMES TO CHCK IOQUE BEFORE CHECKING
377                                     ;TYPE0 WHEN BOTH HAVE REQUESTS PENDING
378 001333 000        MDL45: .BYTE OPEN ;NON-ZERO IF CPU IS 11/45
379 001334 000        GWBFC: .BYTE OPEN ;WRITE BUFFER ROTATION COUNT
380 001335 000        MEMERT: .BYTE OPEN
381 001336 010        RKTIME: .BYTE 10
382 001337 003        ROTM: .BYTE 3 ;# OF GWBUF CALLS (LESS ONE) NEEDED PER
383                                     ;ROTRUF CALL.
384 001340 000        PPRES: .BYTE OPEN ;PARITY PRESENT, 0=INHIBITED OR NONEXISTENT
385 001341 000        CNT1: .BYTE OPEN
386 001342 000        ATE: .BYTE OPEN
387 001343 000        WFLAG: .BYTE OPEN
388 001344 000        HNGFLG: .BYTE OPEN ;FLAG INDICATES CKHNG ROUTINE IS BUSY
389 001345 000        MOVFLG: .BYTE OPEN ;FLAG USED IN MPCODE ROUTINE
390 001346 000        MSGFLG: .BYTE OPEN ;FLAG INDICATING *SGN OR *SGS CALL
391 001347 000        DATFLG: .BYTE OPEN ;FLAG INDICATING CKDATA OP DATABK CALL
392 001350 012        SYSLIM: .BYTE 10. ;MAX. NUMBER OF SYSTEM ERRORS ALLOWED
393 001351 000        TFLAG: .BYTE OPEN ;TIMER FLAG INDICATING ERROR OR EOP
394 001352 000        NSTOP: .BYTE OPEN ;FLAG PREVENTING STOP FROM BEING SET
395 001353 000        CLOCK: .BYTE OPEN ;CLOCK FLAG
396 001354 000        ALLBK: .BYTE OPEN ;WHEN NON 0 MEANS RUN ALL BKMODS BEFORE RELOCATING
397 001356 000000G      .EVEN
398 001356 000000G      CLOCKL: KW11L ;WILL BE 0 IF NO KW11=L CLOCK
399 001360 000000G      CLOCKP: KW11P ;WILL BE 0 IF NO KW11=P CLOCK
400
401 001362 001604      HNGTIM: 900. ;HUNG TIME LIMIT IN SECONDS
402 001364 000000G      ALCLR: LCLEAR ;SUBROUTINE ADDRESS IN L CLOCK MODULE
403 001366 000000G      APCLR: PCLEAR ;SUBROUTINE ADDR IN P CLOCK MODULE
404                                     ;WILL DROP ANY I/O MODULE INACTIVE FOR 15. MINS.
  
```

```

405 ; *** DO NOT CHANGE THE SEQUENCE OF THE NEXT 9 WORDS *** !
406
407 ;REFER TO THE DOCUMENT FOR THE MONITOR CHECKSUM MODULE (BKBA)
408 ;FOR THE USE OF THE NEXT SIX WORDS, AND HOW THE TWO "OPEN" LOCATIONS
409 ;ARE DEFINED WHEN A NEW VERSION OF THE MONITOR IS RELEASED
410
411 001370 137577 CHK2: 137577 ;LOW CHKSUM FOR PROC. WITH NO RTT INST.
412 001372 002641 CHK2: 2641 ;HIGH CHKSUM
413 001374 001534 CHKSUM: ,WORD START ;BEGIN ADDR OF MONITOR PURE AREA
414 001376 025106 ,WORD WCASEE ;END ADDR OF MONITOR PURE AREA
415 001400 137607 CHK1: 137607 ;CONTAINS LOW CKSM OF MONITOR PURE AREA (PROC WITH RTT)
416 001402 002641 CHK1: 2641 ;CONTAINS HI CKSM OF MONITOR PURE AREA
417 ; ,ODD
418 001405 000003 HOURS: ,BLKB 3 ;HOURS IN ASCII
419 001410 000000 MINITS: OPEN ;MINUTES IN ASCII
420 001412 000000 SEKNS: OPEN ;SECONDS IN ASCII
421 001414 000000 RTIME: OPEN ;TOTAL ELAPSED RUNTIME IN SECONDS
422 001416 000000 RTIMX: OPEN ;HOLDS RUNTIME EXTENDED BITS
423 001420 000000 CLKADR: OPEN ;SAVE LOCATION FOR CLOCK ADDRESS
424 001422 000000 CLKHLD: OPEN ;SAVE LOCATION FOR CLOCK CSR
425 001424 000000 PTIME: OPEN ;HOLDS MODULE PASS TIME IN SECONDS
426 001426 000000 MAXTIM: OPEN ;HOLDS MAX. ELAPSED TIME FOR CKHUNG ROUTINE
427 001430 000020 PARTAB: ,BLKB 16. ;TABLE OF PARITY REGISTER ADDRESSES
428 001470 000000 CPUERR: OPEN ;ADDRESS OF 11/60 CPU ERROR REG.
429 001472 000000 MEMERR: OPEN ;ADDRESS OF 11/60 MEMORY ERROR REG.
430 001474 000000 LGHOLD: OPEN ;HOLDS CONTENTS OF ERROR LOG REG. LAST READ/WRTTEN
431 001476 177746 CONTRL: 177746 ;CACHE CONTROL REG
432 001500 000000 KONTRL: 0 ;TEMP STOR FOR CACHE OPERATIONS
433
434 ;.EVEN
    
```

```

435 ;VARIABLE QUE CALL AREA- KEPT IN FRONT OF START SINCE IT IS IMPURE CODE
436
437 ;COMMON "QUE CALL" ROUTINE, DOES A DIRECT DISPATCH RATHER THAN
438 ;REQUEING DUE TO MULTIPLE POINTS CALLING THIS ROUTINE WHICH
439 ;MAY OVERLAP IF REQUEUED.
440
441 001502 012700 001512 CONQUE: MOV #CADDR,RO
442 001506 000167 001304 ; JUMP TO I0GSVN
443 001512 000000 CADDR: OPEN ;DESTINATION ADDR.
444 001514 000000 CSTART: OPEN ;MODULE START ADDR. (0 FOR MONITOR).
445
446 ;CHECKDATA ERROR CALL
447 001516 104421 CDERR: DERR: ;
448 001520 000000 CDERR: OPEN ;ADDRESS OF CKDATA CALL
449 001522 000167 006736 ; JUMP TO CDRET
450
451 ;TELETYPE ROUTINE PIRG CALL
452 ;TTYLNK IS FREQUENTLY MODIFIED DURING EXECUTION
453
454 001526 000004 TTYSRV: PIRG:
455 001530 007436 TTYLNK: TTSRV:
456 001532 000000 ;
    
```

```

;STARTUP SEQUENCE
457
458
459 001534 000005          START:  RESET          ;CLEAR THE WORLD.
460 001536 012706 027212      MOV          #SPROT,R6      ;SET UP STACK.
461 001542 012767 015460 176254  MOV          #PWRDN,PWRFY   ;SET UP POWER FAIL VECTOR.
462 001550 012746 000004      MOV          #BUSEV,-(SP)   ;SAVE PRESENT BUS ERR TRAP
463 001554 012767 177570 177414  MOV          #HARDSR,SR     ;SET SP=HARDWARE SR
464 001562 012767 001576 176214  MOV          #36,BUSEV      ;SET NEW BUS ERR TRAP
465 001570 005777 177402      TST         #BR            ;IS IT HARDWARE SR?
466 001574 000404            BR          48              ;IF YES, RESTORE TRAP
467 001576 012767 001004 177372 38:  MOV          #FAKESR,SR     ;ELSE SET UP SOFTWARE SR
468 001604 022626            CMP          (SP)+,(SP)+    ;RESTORE STACK
469 001606 012667 176172 48:  MOV          (SP)+,BUSEV    ;RESTORE BUS ERR TRAP
470 001612 012767 026520 177450  MOV          #MODQ-2,BKPTR  ;INITIALIZE TO START BKNODS WITH 1ST LOC
471 001620 005767 177374      TST         LLOCORE        ;DOME BUFFER SETUP?
472 001624 001054            BNE         28              ;BR IF YES.
473 001626 004567 025420      JSR         R5,SETBUF      ;NO, DO IT.
;THE FOLLOWING IS USED TO WRITE ALL MEMORY WITH WHATEVER IS
;ALREADY THERE, JUST TO CLEAR ANY PARTTY ERRORS PRESENT
474
475
476 001632 000005          RESET          ;TURN OFF KT
477 001634 105767 177444      TSTB       KTPRES         ;KT?
478 001640 001430            BEQ         14             ;BR IF NO
479 001642 004767 010146      JSR         PC,KTINIT     ;SET UP KT REG'S FOR NORMAL USE
480 001646 005067 170432      CLR        KIPDR2        ;FOPCK A KT TRAP WHEN LOOP OVERFLOWS
481
482 001652 012737 000001 177572  MOV          #1,##SSRO     ;TURN ON KT
483 001660 012737 001736 000004  MOV          #178,##4      ;EXIT POINT FOR END OF MEMORY
484 001666 012701 177600      MOV          #-200,R1      ;SET BACK A PAGE FOR LOOP'S INC
485 001672 012737 001700 000250  MOV          #138,##250    ;SET UP KT TRAP TO RUMP TO NEXT PAGE
486 001700 062701 000200 138:  ADD         #200,R1        ;BUMP TO NEXT PAGE
487 001704 010167 170432      MOV          R1,KIPAR1     ;SET UP LOOP'S PAGE
488 001710 012706 001160      MOV          #CCSRA,R6     ;GET PID OF TRAP JUNK ON STACK
489
490
491 001714 012700 020000          MOV          #20000,R0     ;USING THIS AREA FOR STACK TEMPORARILY SINCE ITS
492 001720 000404            BR          158           ;IN THE SAME PAR SPACE
493 001722 012737 001744 000004 148:  MOV          #188,##4     ;SET R0 TO USE PAR 1
494 001730 005000            CLR         R0             ;GO DO A PAGE
495 001732 011020 158:  MOV          (R0),(R0)+    ;SET UP EXIT FOR NON KT
496 001734 000776            BR 158                    ;START AT LOC 0
497 001736 012737 000000 177572 178:  MOV          #0,##SSRO    ;WRITE A LOC WITH ITSELF
498 001744 012706 027212 188:  MOV          #SPBOT,R6    ;BR TILL KT OP 4 TRAP
499 001750 012737 015246 000004      MOV          #BUSERR,##4   ;TURN KT BACK OFF
500
501 001756 004767 013004 28:  JSR         PC,CLRQUS     ;RESET STACK TO REAL ARFA
502 001762 105067 177323      CLR        RMODE         ;CORRECT TRAP VECTOR
503 001766 112767 177777 177025  MOV          #-1,MDXCTR    ;CLEAR QUEUES.
504 001774 105067 177314      CLR        ROTL          ;INITIALIZE COUNTER
505 002000 104406 022672      MSG#,TITLE  ;MAKE SURE ROTATION IS NOT LOCKED
506 002004 022767 001004 177164  CMP         #FAKESR,SR    ;TYPE THE TITLE
507 002012 001002            BNE         55#          ;DO WE HAVE A REAL SP?
508 002014 104406 024633      MSG#,NOSWP  ;BR IF YES
509 002020 004767 014714 55:  JSR         PC,POTCHK     ;ELSE TELL HIM WHERE IT IS
510 002024 004767 015022      JSR         PC,POTCHK     ;CHECK IF ROTATE ALLOWED AND TYPE RANGE.
511 002030 004767 015514      JSR         PC,SYSIZ     ;PRINT OUT THE SYSTEM SIZE
512 002034 105767 177244      JSR         PC,KLOCK     ;CHECK FOR A CLOCK AND SET FLAGS
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545

```

```

513 002040 001410          START2: BEQ          START3      ;NO- BRANCH
514 002042 004767 007746      JSR         PC,KTINIT     ;YES- INITIALIZE
515 002046 012737 000001 177572  MOV          #1,##SSRO    ;TURN ON 18-BJT KT11
516 002054 104406 024615      MSG#,KTONM  ;"KT11 ENABLED"
517
518
519 002060 000402          START3: BR          START4      ;CONTINUE
520 002062 104406 023776      MSG#,NOKT  ;"NO KT11"
521
522 002066 004767 010722      START4:  JSR         PC,PDNN  ;GO TURN ON PARITY IF AVAILABLE
523
524 002072 105737 000041          TSTB       ##41          ;ANY LOAD MEDIUM BITS SET ?
525 002076 001402            BEQ         18             ;NO, CONTINUE
526 002100 104406 023520      MSG#,CLR40 ;" TO EXERCISE LOAD MEDIUM YOU MUST CLEAR LOC 40
527 002104 005067 177134 18:  CLR         OFFSTD        ;MAKE SURE PROGRAM IS NOT PARTIALLY RELOCATED
528 002110 004767 017360      JSR         PC,MVCODE     ;CLEAR QUEUES IN CASE MVCODE OVER WROTE THEM
529 002114 004767 012646      JSR         PC,CLRQUS    ;INITIALIZE VECTOR AREA
530 002120 004767 012724      JSR         PC,CLRVEC    ;KT11 IN USE ?
531 002124 105767 177154      TSTB       KTPRES        ;YES, CONTINUE
532 002130 001002            BNE         58             ;NO, LOAD WORST CASE PATTERN IN BUFFER
533 002132 004767 022534 58:  JSR         PC,LOADWC     ;TYPE DOT.
534 002136 104406 022737      MSG#,DOT   ;CHAIN MODE?
535 002142 105767 177144      TSTB       CHN           ;NO- GO AWAIT KEYBOARD INPUT
536 002146 001405            BEQ         #15,#KBPTR   ;CLEAN OUT KEYBOARD BUFFER WITH A CR
537 002150 112777 000015 176660 68:  MOV          #15,#KBPTR  ;YES-START RUNNING.
538 002156 000167 006526      JMP         RUN
539
540 002162 012706 027212      INPUT:  MOV          #SPBOT,R6 ;RESET STACK.
541 002166 105067 177116      CLR        SYSERR        ;CLEAR SYSTEM ERROR INDICATOR.
542 002172 012767 026712 176636  MOV          #KBUF,KRPTR  ;INITIALIZE KYBD BUFFER POINTER.
543 002200 052777 000100 176754  RIS         #IE,#TKS     ;ENABLE KYBD INTERRUPTS.
544 002206 116767 177120 176613  MOV        QUECNT,QUECTR
545

```

```

546 ;QUEUE CHECK. NOTE THAT IF IOBKID IS NEGATIVE, NO TYPE QUE REQUESTS WILL BE SERVICED
547
548 002214 105367 176607 QÜETST: DECB QUECTR ;COUNT TO PREVENT BREAK CALLS FROM LOCKING OUT TYPING
549 002220 003004 BGT 18 ;ALSO PREVENTS HIGH SPEED INT DEVICES FROM
;LOCKING OUT TYPE QUEUE AND RUN SERVICE
550 ;STARTUP OF MODULES
551
552
553 002222 116767 177104 176577 MOVB QÜECNT,QUECTR ;RESET QUE COUNTER
554 002230 000405 BR 28
555 002232 105767 176550 18: TSTB IOQUE ;IO QUE REQUEST PENDING?
556 002236 001402 BEQ 28
557 002240 000167 000476 JMP IOQSVK ;GO SERVICE IF YES.
558 002244 005767 176610 28: TST IOBKID ;STARTING I/O MODULES?
559 002250 100422 BMI QTSTC ;BR IF YES.
560 002252 105767 176531 38: TSTB TYPQUE ;TYPE REQUEST PENDING?
561 002256 001412 BEQ QTSTB ;BR IF NOT.
562 002260 105767 176530 TSTB TTYBSY ;TTY BUSY?
563 002264 001002 BNE QTSTA ;BR IF YES.
564 002266 000167 000674 JMP TYPSTV ;NO. GO SERVICE TYPE QUEUE.
565 002272 005367 176612 QTSTA: DEC TYPCTR ;CHECK FOR TTY HUNG
566 002276 001002 BNE QTSTB
567 002300 000167 005132 JMP TTSRV1 ;UNHANG TTY
568 002304 005767 176560 QTSTB: TST TRCPC ;BACKGROUND MODULE PENDING?
569 002310 001402 BEQ QTSTC
570 002312 000167 000522 JMP BRQSVK ;JUMP IF YES
571 002316 105767 176767 QTSTC: TSTB RMODE ;IN RUN MODE?
572 002322 001734 BEQ QÜETST ;BR IF NOT. GO CHECK AGAIN.
    
```

```

573 ;RUN MODE SERVICE ROUTINE.
574 ;NOTE THAT MODULE'S STACK POINTER IS REINITIALIZED BUT OTHER
575 ;REGISTERS HAVE ANY PREVIOUS VALUES RESTORED AT START OR RESTART
576
577 002324 105767 176463 RUNSVC: TSTB BRAKE ;IS THE BRAKE ON?
578 002330 001331 BNE QÜETST ;BR IF YES. DO NOT INIT MORE MODULES.
579 002332 105767 177016 TSTB ALLBK ;RUN ALL BKMODS BEFORE RELOCATING ?
580 002336 001407 BEQ 128 ;NO, CONTINUE
581 002340 105767 176460 TSTB STOP ;WAITING TO RELOCATE ?
582 002344 001407 BEQ 138 ;NO, CONTINUE
583 002346 012767 040020 176504 MOV #40020,IOBKID ;START UP ONLY BKMOD'S
584 002354 000415 BR 18 ;CONTINUE
585
586 002356 105767 176442 128: TSTB STOP ;STOP SET?
587 002362 001314 BNE QÜFTST ;YES, DON'T START ANY MORE MODULES
588 002364 105767 176426 138: TSTB MODCTR ;MODCTR #0?
589 002370 001007 BNE 18 ;BR IF NOT.
590 002372 105767 176714 TSTB CHN ;YES, ARE WE IN CHAIN MODE?
591 002376 100004 BPL 18 ;BR IF NOT.
592 002400 012767 040020 176452 MOV #40020,IOBKID ;SWITCH IT TO BACKGROUND AND PREVENT LOOPING
593 002406 000702 BR QÜFTST ;THRU BACKGROUND MODULE SERIES MORE THAN ONCE
594 002410 062767 000002 176422 18: ADD #2,MODPTR ;POINT TO NEXT MODULE.
595 002416 017700 176416 MOV @MODPTR,RO ;MODULE ADDR TO RO.
596 002422 001476 BEQ 78 ;BR IF NO ADDR.
597 002424 016446 000020 MOV STAT(RO),-(SP) ;GET MODULE'S STATUS
598 002430 042716 016000 BIC #BIT12|BIT11|BIT10,(SP) ;MASK SPECIAL INDICATORS
599 002434 022667 176420 CMP (SP)+,IOBKID ;CORRECT MODULE TYPE TO RUN?
600 002440 001136 BNE 118 ;NO- BRANCH
601 002442 005767 176574 TST OFFSET ;IF OFFSET 0, IGNORE BITS 10 +11
602 002446 001414 BEQ 28 ;CONTINUE
    
```

```

603 002450 032760 004000 000020      BIT      #BIT11,STAT(RO) ;BIT11 SET?
604 002456 001127                BNE      11$          ;YES, DON'T RUN MODULE SINCE OFFSET NOT 0
605 002460 032767 001777 176554      BIT      #1777,OFFSET ;OFFSET 32K MULTIPLE?
606
607 002466 001404                BEQ      2$          ;YES- IGNORE BIT 10
608 002470 032760 002000 000020      BIT      #BIT10,STAT(RO) ;NO- BIT10 SET?
609 002476 001117                BNE      11$          ;YES- DON'T RUN MODULE
610 002500 005767 176354      28:     TST      IOBKID ;BACKGROUND MODULE?
611
612
613
614 002504 100410                BMI      3$          ;BR IF NOT,
615 002506 112767 177777 176277      MOVB    #-1,BRAKE ;YES, APPLY BRAKE,
616 002514 016767 176320 176546      MOV     MODPTR,BKPTR ;RECORD MODQ ADDRESS OF CURRENT BKM0D
617 002522 004767 015314                JSR     PC,BKTYM ;RECORD START TIME OF THIS BKM0D
618 002526 032760 010000 000020 38:     BIT      #BIT12,STAT(RO) ;EXTENDED I/O MODULE?
619 002534 001402                BEQ     4$          ;NO- BRANCH
620 002536 105267 176260                INCB   MDXCNT ;YES, COUNT IT
621 002542 105060 000005                CLRB  XFLAG(RO) ;CLEAR INDICATOR
622 002546 016060 000024 000046      MOV     SPOINT(RO),SVR6(RO) ;SET UP MODULE SP POINTFR,
623 002554 105767 176546                TSTB  RUNRFL
624 002560 001004                BNE     5$          ;
625 002562 016067 000022 176264 44:     MOV     INIT(RO),DSTADR ;SET UP DESTINATION ADDR,
626 002570 000407                BR     6$          ;
627 002572 016067 000062 176254 58:     MOV     RSTRT(RO),DSTADR
628 002600 005760 000026                TST   PSCNT(RO) ;HAS MODULE BEEN THRU START YET?
629 002604 001001                BNE     6$          ;BR IF YES
630 002606 000765                BR     44$         ;ELSE GO BACK
631 002610 105367 176202      66:     DECB   MODCTR ;DECR COUNT OF MODS INITED,
632 002614 000167 000212                JMP   IOQSVL ;GO TO START MODULE
633 002620 022767 040000 176232 78:     CMP     #40000,IOBKID ;FINISHED WITH SPECIAL BKM0D ?
634 002626 001004                BNE     8$          ;BR IF NOT,
635 002630 012767 041000 176222      MOV     #41000,IOBKID ;YES, SWITCH TO NBKM0D
636 002636 000434                BR     10$         ;CONTINUE
637 002640 022767 041000 176212 88:     CMP     #41000,IOBKID ;FINISHED WITH NBKM0DS ?
638 002646 001013                BNE     9$          ;NO, CONTINUE
639 002650 012767 140000 176202      MOV     #140000,IOBKID ;YES, SWITCH TO IQM0D,
640 002656 112767 177777 176135      MOVB   #-1,MDXCTR ;INITIALLY LOCK MDXCTR
641
642
643
644
645 002664 105067 176132                CLRB   MDXCNT ;INITIALIZE COUNT
646
647
648
649
650
651 002670 105067 176456                CLRB   NSTOP ;ALLOW STOP FLAG TO BE SET
652 002674 000415                BR     10$         ; CONTINUE
653 002676 005767 176156      98:     TST   IOBKID ;IF ALREADY BACKGROUND
654 002702 100012                BPL   10$         ;DON'T RESET MDXCTR
655
656
657
658
    
```

```

659 002704 116767 176112 176107      MOVB   MDXCNT,MDXCTR ;SETUP MDXCTR
660 002712 012767 040020 176140      MOV     #40020,IOBKID ;SWITCH TO BACKGROUND MODE
661
662
663
664
665 002720 016767 176344 176112      MOV     BKPTR,MODPTR ;CONTINUE WITH NEXT BKM0D
666                                MOVB   #1,RUNRFL ;SET RUN RESTART FLAG11/60 *****
667 002726 000403                BR     11$
668 002730 012767 026520 176102 108:     MOV     #MODQ-2,MODPTR ;POINT TO MODULE TABLE START,
669 002736 000167 177252      118:     JMP   QUETST
    
```

```

670 ;I/O QUE SERVICE ROUTINE.
671 ;SERVICES PIRQ,QUES, AND BREAKS CALLS
672 ;NOTE THAT PIRQ SERVICE CAN OCCUR ON TOP OF THIS ROUTINE- SO
673 ;LEVEL 7 PRIORITY IS REQUIRED IF COUNT IS DECREMENTED AFTER IOQ2
674 ;IS MOVED
675
676 002742 026727 176130 030072 IOQSVCI CMP IOQ2,#IOQLIM ;REACHED LIMIT OF QUEUE?
677 002750 103403 BLO 18 ;BR IF NOT.
678 002752 012767 027252 176116 MOV #IOQ,IOQ2 ;RESET IOQ2.
679 002760 105367 176022 18: DECB IOQUE ;DECREMENT REQUEST COUNT
680 002764 017700 176106 MOV #IOQ2,R0 ;GET PC+2 OF CALL.
681 002770 062767 000002 176100 ADD #2,IOQ2 ;UPDATE IOQ2.
682 002776 010001 MOV R0,R1 ;SAVE PC+2 OF CALL
683 003000 014046 MOV -(R0),-(SP) ;PUSH CALL ON STACK
684 003002 022726 104407 CMP #BREAK,(SP)+ ;BREAK CALL?
685 003006 001002 BNE 28 ;NO, CONTINUE
686 003010 000167 000440 JMP BREAK. ;YES, GO SERVICE IT
687 003014 005720 28: TST (R0)+
688
689 003016 012067 176032 IOQSVNI MOV (R0)+,DSTADR ;GET DESTINATION ADDR.
690 003022 012000 MOV (R0)+,R0 ;GET MODULE ADDR. IS IT 0?
691 003024 001002 BNE IOQSVI ;BR IF NOT. IT'S A MODULE.
692 003026 000177 176022 JMP @DSTADR ;GO DO MONITOR FUNCTION.
693 003032 116067 000020 176026 IOQSVI: MOVE STAT(R0),RSTAT ;GET RUN STATUS.
694 003040 032760 020000 000020 IOQSVAI: BIT #BIT13,STAT(R0) ;MODULE STOPPED?
695 003046 001402 BEQ 18 ;NO, CONTINUE
696 003050 000167 177140 JMP QUETST ;YES, FORGET IT
697 003054 032760 040000 000020 18: BIT #BIT14,STAT(R0) ;IS MODULE SELECTED ?
698 003062 001002 BNE 28 ;YES, CONTINUE
699 003064 000167 177124 JMP QUETST ;NO, FORGET IT
700 003070 010067 176042 28: MOV R0,MODADR ;SAVE R0. (MODULE ADDR).
701 003074 010667 176052 MOV R6,SPSAV ;SAVE MONITOR STACK TO01.
702 003100 062700 000050 ADD #SVR6+2,R0 ;RESTORE MODULE'S REGS.
703 003104 014006 MOV -(R0),R6 ;STARTING WITH STACK POINTER.
704 003106 014005 IOQSVDI: MOV -(R0),R5
705 003110 014004 MOV -(R0),R4
706 003112 014003 MOV -(R0),R3
707 003114 014002 MOV -(R0),R2
708 003116 014001 MOV -(R0),R1
709 003120 014000 MOV -(R0),R0
710 003122 016746 175740 IOQSVBI: MOV RSTAT,-(SP) ;LOAD RUN STATUS.
711 003126 105066 000001 CLR B ;KERNEL, 1ST REGISTR SET
712 003132 016746 175716 MOV DSTADR,-(SP) ;LOAD DESTINATION ADDR.
713 003136 000006 RTTI: RTT ;GO TO DESTINATION.
    
```

```

714 ;BACKGROUND QUEUE SERVICED HERE.
715
716 003140 016767 175724 175706 BKQSVCI: MOV TRCPC,DSTADR ;SET UP DESTINATION ADDR.
717 003146 116767 175720 175712 MOV TRCPSW,RSTAT ;SET UP RUN STATUS.
718 003154 017700 175660 MOV #MODPTR,R0 ;MODULE START ADDR TO R0.
719 003160 005067 175704 CLR TRCPC ;CLEAR BK MODULE WAITING INDICATOR.
720 ;IF ERROR OCCURS, WILL STOP MODULE FROM FURTHER
721 ;EXECUTION UNTIL ERROR IS REPORTED
722 003164 000725 BR IOQSVI ;GO GET GOING.
    
```

```

723 ;TYPE QUE SERVICE ROUTINE.
724 ;REMOVES PC+2 OF CALL FROM Q, PASSING IT IN R1
725
726 003166 026727 175710 031222 TYP&VC: CMP TYPQ2,#TYPLIM ;REACHED UPPER END OF QUEUE?
727 003174 103403 BLO 18 ;BR IF NOT.
728 003176 012767 030072 175676 MOV #TYPEQ,TYPQ2 ;YES, RESET TYPQ2.
729 003204 105267 175604 18: INCB TTYBSY ;INDICATE TTY BUSY.
730 003210 017701 175666 MOV @TYPQ2,R1 ;GET PC+2 OF CALL.
731 003214 062767 000002 175660 ADD #2,TYPQ2 ;UPDATE TYPQ2.
732 003222 105367 175561 DECB TYPQUE ;DECREMENT REQUEST COUNT.
733 003226 010146 MOV R1,-(SP) ;PUSH PC+2 OF CALL
734 003230 005741 TST -(R1) ;POINT TO CALL.
735 003232 010167 175610 MOV R1,NUMBER ;SAVE ADDRESS OF CALL
736 003236 011101 MOV (R1),R1 ;GET CALL
737 003240 006301 ASL R1 ;TIMES 2.
738 003242 016101 172244 MOV TYPTAB-TRP2-4(R1),R1 ;FORM SERVICE ADDR.
739 003246 000201 RTS R1 ;GO TO IT, RESTORE R1.
740
741 003250 005310 TYPTAB: .WORD PAGEEND
742 003252 005014 .WORD ENDSVC,ERRSVC,ERSVC1
743 003260 003426 000001 004024 .WORD MSG,,1,ERSVC2,MSGN,,1,1,1,1,MSG,,1,1,DERRX,,MSGD.
744 003266 003322 000001 000001
745 003274 000001 000001 003312
746 003302 000001 000001 003562
747 003310 003466
    
```

```

748 ;MSGN, ROUTINE, SERVICES CALLS TO TYPE ASCII MESSAGES.
749
750 003312 112767 177777 176026 MSGS,: MOVB #=1,MSGFLG ;SET MSGS CALL FLAG
751 003320 000402 BR MSCONT ;CONTINUE
752 003322 105067 176020 MSGN,: CLRB MSGFLG ;SET MSGN CALL FLAG
753
754 003326 012167 175532 MSCONT: MOV (R1)+,TABADR ;GET ASCII TABLE ADDR.
755 003332 012167 176156 MOV (R1)+,CSTART ;MODULE ADDR TO CSTART.
756 003336 010167 176150 MOV R1,CADDR ;RESUME ADDR TO CADDR.
757 003342 016701 176146 MOV CSTART,R1 ;MODULE ADDR TO R1.
758 003346 032777 020000 175622 BIT #BIT13,4SR ;INHIBIT ERROR PRINT?
759 003354 001033 BNE MSG1 ;BR IF YES.
760 003356 105767 175764 TSTB MSGFLG ;IS THIS A MSGS CALL?
761 003362 100410 BMI 18 ;YES, CONTINUE
762 003364 004767 001300 JSR PC,ENDCOM ;NO, DO COMMON STUFF.
763 003370 004567 003754 JSR R5,TYPE ;TYPE
764 003374 022744 CRCR ;? CARRAGE RETURNS
765 003376 025721 AEND ;COMMON HEADER.
766 003400 025756 $PASS ;PASS #.
767 003402 000000 0
768 003404 016701 175454 18: MOV TABADR,R1 ;TABLE ADDR TO R1.
769 003410 012146 28: MOV (R1)+,-(SP) ;GET MESSAGE ADDR.
770 003412 022716 177777 CMP #=1,(SP) ;TERMINATOR?
771 003416 001412 BEQ MSG1 ;BR IF YES, DONE.
772 003420 004767 003742 JSR PC,TYPE ;NO, TYPE MESSAGE.
773 003424 000771 BR 28 ;GO DO IT AGAIN.
    
```

774									
775									
776	003426	012146		MSG.:	MOV	(R1)+,(SP)			;ASCII MESSAGE ADDR TO STACK.
777	003430	010167	176056		MOV	R1,CADDR			;RESUME ADDR TO CADDR.
778	003434	005067	176054		CLR	CSTART			;INDICATE MONITOR QUE CALLING.
779	003440	004767	003722		JSR	PC,TYPE			;GO TYPE DESIRED MESSAGE.
780	003444	105067	175344	MSG1.:	CLRB	TIYBSY			
781	003450	000167	176026		JMP	COMQUE			;GO QUEUE UP TO RESUME.

782									
783									
784	003454	012100							
785	003456	010167	175372	BREAK.:	MOV	(R1)+,R0			;GET MODULE ADDR.
786	003462	000167	177344		MOV	R1,DSTADR			;GET DESTINATION ADDR.
					JMP	IQQSVE			;QUE UP TO RESUME

```

787 ;MSGD8 SERVICE- TYPE DATA ERROR COUNT AND TRANSFER SIZE
788 ;AND HALT MODULE IF NECESSARY
789
790 003466 010167 176020 MSGD1: MOV R1,CADDR ;SET UP RESUME ADDRESS
791 003472 004767 001254 JSR PC,TYPDAT ;GET MODULE ADDRESS
792 003476 012601 MOV (SP)+,R1
793 003500 010167 176010 MOV R1,CSTART ;SAVE ADDRESS
794 003504 004567 010124 JSR RS,FILLNM ;GET MODULE NAME
795 003510 026424 AERNM+1
796 003512 004767 001234 JSR PC,TYPDAT ;SETUP ERROR COUNT (ASCII)
797 003516 004567 010650 JSR RS,BDCNV
798 003522 026436 AERCT
799 003524 004767 001222 JSR PC,TYPDAT ;SETUP TOTAL WORDS TRANSFERRED
800 003530 004567 010636 JSR RS,BDCNV
801 003534 026470 AERTOT
802 003536 032777 020000 175432 BIT #BIT13,#SR ;INHIBIT ERROR PRINTOUT?
803 003544 001004 BNE MSGD1 ;YES- BRANCH
804 003546 012746 026423 MOV #AERNM,-(SP) ;NO- TYPE MESSAGE
805 003552 004767 003610 JSR PC,TYPE
806 003556 000167 001014 MSGD1: JMP ERSVND ;CHECK FOR HALTING MODULE, AND RETURN IF NOT
    
```

```

807 ;DERRX. ROUTINE. SERVICES EXTENDED DATA ERROR CALLS FROM THE MONITOR
808 ;CHECKDATA ROUTINE. TYPES ERROR INFORMATION AND RETURNS
809
810 003562 112767 000001 175227 DERRX: MOV# #1,ERRIND
811 003570 005721 TST (R1)+ ;POINT TO RETURN LOCATION
812 003572 010167 175714 MOV R1,CADDR ;SETUP RESUME ADDRESS
813 003576 004767 001150 JSR PC,TYPDAT ;GET MODULE ADDRESS
814 003602 012601 MOV (SP)+,R1
815 003604 010167 175704 MOV R1,CSTART ;AND SAVE IT
816 003610 004767 001136 JSR PC,TYPDAT
817 003614 012667 175226 MOV (SP)+,NUMBER ;SETUP ADDRESS OF CKDATA CALL
818 003620 004767 001044 JSR PC,ENDCOM
819 003624 012702 000005 MOV #5,R2
820 003630 004767 001116 DERRX1: JSR PC,TYPDAT
821 003634 005302 DEC R2
822 003636 001374 BNE DERRX1
823 003640 016146 000030 MOV ERCNT(R1),-(SP) ;ERROR COUNT TO STACK
824 003644 004567 010522 JSR RS,BDCNV ;CONVERT ERROR COUNT TO DECIMAL
825 003650 026335 AERNMB
826 003652 005046 CLR -(SP)
827 003654 005766 TST 2(SP)
828 003660 001402 BEQ 16
829 003662 012716 000060 MOV #60,(SP) ;CSRA EA BITS
830 003666 004567 010260 16: JSR RS,OACNVX ;CSR ADDRESS- ASCII
831 003672 026205 ADTE6
832 003674 012603 MOV (SP)+,R3 ;SAVE TABLE ADDRESS
833 003676 012667 175252 MOV (SP)+,ERRPOS ;SAVE OFFSET COUNT
834 003702 006367 175246 ASL ERRPOS
835 003706 012346 MOV (R3)+,-(SP) ;SETUP READ BUFFER PA
836 003710 012346 MOV (R3)+,-(SP) ;GET EA BITS
837 003712 066766 175236 000002 ADD ERRPOS,2(SP)
838 003720 103002 BCC 2#
839 003722 062716 000020 ADD #20,(SP)
840 003726 005723 26: TST (R3)+ ;SKIP SIZE
841 003730 004567 010216 JSR RS,OACNVX ;GET PA OF RDADR
842 003734 026265 ADTE4
843 003736 012346 MOV (3)+,-(SP) ;GET WRITE PA
844 003740 012346 MOV (3)+,-(SP) ;GET EA BITS
845 003742 066766 175206 000002 ADD ERRPOS,2(SP)
846 003750 103002 BCC 3#
847 003752 062716 000020 ADD #20,(SP)
848 003756 004567 010170 36: JSR RS,OACNVX ;GET ASCII OF WRADR
849 003762 026250 ADTE5
850 003764 004567 010150 JSR RS,OACNV ;CONVERT S/B TO ASCII
851 003770 026220 ADTE3
852 003772 004567 010142 JSR RS,OACNV ;CONVERT WAS TO ASCII
853 003776 026233 ADTE2
854 004000 016746 175150 MOV ERRPOS,-(SP) ;ERROR POSITION IN BUFFER
855 004004 000241 CLC ;CORRECT BACK FROM RYTE TO WORD PTR
856 004006 006216 ASR (SP)
857 004010 003216 INC (SP) ;MAKE 1ST WORD #1
858 004012 004567 010354 JSR RS,BDCNV ;GET DECIMAL
859 004016 026317 ADTE7
860 004020 000167 000306 JMP ERSVX ;GO TO COMMON ROUTINE
    
```

```

;ERROR CALLS ARE SERVICED HERE,
;R1 AT ENTRY CONTAINS PC+2 OF CALL
061
062
063
064 004024 112767 000003 174765 ERVVC2: MOV# #3,ERRIND ;INDICATE ERRORRN CALL.
065 004032 000406 BR ERVCA ;INDICATE DATA ERROR.
066 004034 105067 174757 ERVVC1: CLRB ERRIND
067 004040 000403 BR ERVCA
068 004042 112767 000002 174747 ERRSVC: MOV# #2,ERRIND ;INDICATE "NORMAL" ERROR.
069 004050 012167 175440 ERVCA: MOV (R1)+,CSTART ;SAVE START ADDR OF MODULE.
070 004054 122767 000003 174735 CMPB #3,ERRIND ;ERRORRN CALL?
071 004052 001002 BNE 18 ;BR IF NOT
072 004064 012167 174774 MOV (R1)+,TABADR ;SAVE TABLE ADDR.
073 004070 010167 175416 18: MOV R1,CADDR ;RESUME ADDR TO PSENOB.
074 004074 016701 175414 MOV CSTART,R1 ;GET BACK START ADDR.
075 004100 004767 000564 JSR PC,ENDCOM ;DO COMMON STUFF.
076 004104 005261 000030 INC ERVCTR ;INCREMENT MODULE'S ERROR COUNT.
077 004110 001002 BNE ERVCTR(R1) ;BR IF RESULT NOT 0.
078 004112 005161 000030 COM ERVCTR ;RESET COUNT TO -1.
079 004116 012702 000005 ERVVCB: MOV #5,R2 ;GET TYPE DATA FROM QUEUE TO STACK.
080 004122 004767 000624 ERVCC: JSR PC,TYPDAT ;DO IT.
081 004126 005302 DEC R2 ;DONE?
082 004130 001374 BNE ERVCC ;BR IF NOT.
083 004132 016146 000030 MOV ERVCTR(R1),-(SP) ;ERROR COUNT TO STACK.
084 004136 004567 010230 JSR R5,BDCNV ;CONVERT ERROR COUNT TO DECIMAL.
085 004142 026335 AERNMB
086 004144 105767 174647 TSTB ERRIND ;DATA ERROR?
087 004150 001047 BNE 28 ;BR IF NOT.
088 004152 004567 007762 JSR R5,OACNV ;CONVERT WAS TO ASCII.
089 004156 026233 ADTE2
090 004160 004567 007754 JSR R5,OACNV ;CONVERT S/B TO ASCII,RESET TYPQ2.
091 004164 026220 ADTE3
092 004166 012667 174654 MOV (SP)+,NUMBER ;GET ADDR OF LOCATION READ
093 004172 104413 001046 GETPA,NUMBER
094 004176 016746 174646 MOV NUMBPA,-(SP)
095 004202 016746 174644 MOV NUMBEA,-(SP)
096 004206 004567 007740 JSR R5,OACNVX ;CONVERT WASADR TO ASCII.
097 004212 026265 ADTE4
098 004214 012667 174626 MOV (SP)+,NUMBER
099 004220 104413 001046 GETPA,NUMBER
100 004224 016746 174620 MOV NUMBPA,-(SP)
101 004230 016746 174616 MOV NUMBEA,-(SP)
102 004234 004567 007712 JSR R5,OACNVX ;CONVERT SBADR TO ASCII.
103 004240 026250 ADTE5
104 004242 005046 CLR -(SP) ;ONTO STACK
105 004244 016667 000002 174706 MOV 2(SP),CCSRA ;SAVE CSR ADDRESS
106 004252 001402 BEQ 18 ;IF 0, BRANCH
107 004254 012716 000060 MOV #60,(SP) ;SETUP EA BITS
108 004260 004567 007666 JSR R5,OACNVX ;CONVERT CSR ADDR TO ASCII.
109 004264 026205 ADTE6
110 004266 000421 BR ERVX ;SKIP WAS AND S/B.
111 004270 022626 CMP (SP)+,(SP)+ ;CONVERT STAT REG CONTENTS TO ASCII.
112 004272 004567 007642 JSR R5,OACNV
113 004276 026167 ASTATC
114 004300 004567 007634 JSR R5,OACNV ;CONVERT CSR CONTENTS TO ASCII.
115 004304 026152 ACSRC
116 004306 005046 CLR -(SP) ;ONTO STACK
    
```

```

117 004310 016667 000002 174642 MOV 2(SP),CCSRA ;SAVE CSR ADDRESS
118 004316 001402 BEQ 38 ;BRANCH IF ADDRESS 0
119 004320 012716 000060 MOV #60,(SP)
120 004324 004567 007622 38: JSR R5,OACNVX ;CONVERT CSR ADDR TO ASCII.
121 004330 026136 ACBRAC
122 004332 032777 020000 174636 ERVX: BIT #BIT13,#SR ;INHIBIT ERROR PRINT?
123 004340 001116 BNE ERVND ;BR IF YES.
124 004342 105767 175005 TSTB CLOCK ;IS THERE A CLOCK AVAILABLE ?
125 004346 001414 BEQ 78 ;NO, CONTINUE
126 004350 004767 013476 JSR PC,ERRTYM ;YES, SETUP TIME OF THE ERROR
127 004354 004567 002770 JSR R5,TYPEN ;TYPE
128 004360 022744 CRCR ;2 CARRAGE RETURNS
129 004362 026066 TOTIM ;THE TOTAL TIME
130 004364 026111 PSTIM ;THE PASS TIME
131 004366 025721 AEND ;COMMON HEADER
132 004370 025756 #PASS ;PASS NUMBER
133 004372 026327 ERRNMB ;ERROR NUMBER
134 004374 000000 0 ;THAT'S ALL
135 004376 000407 BR 88 ;CONTINUE
136 004400 004567 002744 78: JSR R5,TYPEN ;TYPE
137 004404 022744 CRCR ;2 CARRAGE RETURNS
138 004406 025721 AEND ;COMMON HEADER.
139 004410 025756 #PASS ;PASS #.
140 004412 026327 ERRNMB ;ERROR #.
141 004414 000000 0
142 004416 122767 000002 174733 88: CMPB #2,ERRIND ;DATA ERROR?
143 004424 003406 BLE 18 ;BR IF NOT.
144 004426 004567 002716 JSR R5,TYPEN ;YES, TYPE
145 004432 026275 ADTE4A ;DATA ERROR VALUES
146 004434 026177 ADTERR
147 004436 000000 0
148 004440 000404 BR 28
149 004442 012746 026130 18: MOV #AERROR,-(SP) ;TYPE ERROR MESSAGE.
150 004446 004767 002714 JSR PC,TYPE
151 004452 122767 000001 174337 28: CMPB #1,ERRIND ;EXTENDED DATA ERROR?
152 004460 001004 BNE 38 ;NO- BRANCH
153 004462 012746 026311 MOV #ADTEX,-(SP) ;YES- OUTPUT POSITION IN BUFFER
154 004466 004767 002674 JSR PC,TYPE
155 004472 012746 022772 38: MOV #ACRLF,-(SP)
156 004476 004767 002664 JSR PC,TYPE
157 004502 122767 000003 174307 CMPB #3,ERRIND ;ERRORRN CALL?
158 004510 001032 BNE ERVND ;BR IF NOT.
159 004512 016702 174346 MOV TABADR,R2 ;TABLE ADDR TO R2.
160 004516 012703 000010 48: MOV #8,,R3 ;WILL TYPE 8 VALUES PER LINE.
161 004522 022712 177777 58: CMP #-1,(2) ;TERMINATOR?
162 004526 001417 BEQ 68 ;BR IF YES, DONE.
163 004530 013246 MOV @2+,-(SP) ;PUT VALUE IN STACK.
164 004532 004567 007402 JSR R5,OACNV ;CONVERT IT TO OCTAL.
165 004536 025675 HOLD6 ;AND PUT IT HERE
166 004540 012746 025675 MOV #HOLD6,-(SP) ;TYPE IT.
167 004544 004767 002616 JSR PC,TYPE
168 004550 005303 DEC R3 ;DONE 8 PER LINE?
169 004552 001363 BNE 58 ;BR IF NOT.
170 004554 012746 022772 MOV #ACRLF,-(SP) ;OUTPUT CRLF.
171 004560 004767 002602 JSR PC,TYPE
172 004564 000754 BR 48 ;GO FOR MORE.
    
```

973	004566	012746	022772	68:	MOV	#ACRLF,-(SP)	;OUTPUT CRLF.
974	004572	004787	002570		JSR	PC,TYPE	
975	004576	005777	174374	ERSVND:	TST	#SR	;HALT MODULE ON ERROR?
976	004602	100410			BMI	18	;BR IF YES.
977	004604	026761	174444	000030	CMP	ERRLM,ERCNT(R1)	;ERROR COUNT 20 OR GREATER?
978	004612	003010			BGT	28	;BR IF NOT, CONTINUE MODULE EXECUTION.
979	004614	032777	040000	174354	BIT	#BIT14,#SR	;YES, HALT MODULE AFTER 20 ERRORS?
980	004622	001004			BNE	28	;NO- SKIP
981	004624	005067	174330	18:	CLR	CCSRA	;CLEAR CCSRA
982	004630	000167	000166		JMP	ENDSVA	;YES, GO HALT MODULE
983	004634	105067	174154	28:	CLRB	TTYBSY	;QUE TO RESUME
984	004640	005767	174314		TST	CCSRA	;IF NOT SETUP, DON'T RESTORE INTO MODULE
985	004644	001407			BEG	38	
986	004646	016700	174642		MOV	CSTART,R0	
987	004652	016760	174302	000050	MOV	CCSRA,CSRA(R0)	;RESTORE CSRA IN MODULE
988	004660	005067			CLR	CCSRA	
989	004664	000167	174612	38:	JMP	COMQUE	

990	004670	104413	001046	ENDCOM:	GETPAS,NUMBER		;NUMBER CONTAINS ADDRESS OF CALL
991	004674	016746	174150		MOV	NUMRPA,-(SP)	
992	004700	016746	174146		MOV	NUMREA,-(SP)	
993	004704	016746	174136		MOV	NUMRER,-(SP)	
994	004710	160116			SUB	R1,(SP)	;COMPUTE ASSEMBLY PC.
995	004712	004567	007222		JSR	R5,OACNV	;CONVERT ASSEMBLY PC TO ASCII.
996	004716	025746			AEND2		
997	004720	004567	007226		JSR	R5,OACNVX	;CONVERT PC TO ASCII.
998	004724	025733			AEND1		
999	004726	004567	006702		JSR	R5,FILLNM	;LET'S GET MODULE NAME.
1000	004732	025721			AEND		;STUFF AT AEND.
1001	004734	016146	000026		MOV	PSCNT(R1),-(SP)	;GET PASS COUNT.
1002	004740	005216			INC	(SP)	;UP IT 1.
1003	004742	004567	007424		JSR	R5,RDCNV	;CONVERT IT TO DECIMAL ASCII.
1004	004746	025766			APASS		
1005	004750	000207			RTS	PC	;LET'S GET OUT.

```

1006 ;TYPDAT ROUTINE LOADS QUEUED DATA ONTO STACK.
1007
1008 004752 011646 TYPDAT: MOV (SP),-(SP) ;SAVE EXIT ON STACK AGAIN.
1009 004754 026727 174122 031222 CMP TYPQ2,#TYPLIM ;REACHED END OF QUEUE?
1010 004762 103403 BLO 1$ ;BR IF NOT.
1011 004764 012767 030072 174110 MOV #TYPEQ,TYPQ2 ;YES, POINT TO START OF QUEUE.
1012 004772 017766 174104 000002 1$ MOV #TYPQ2,2(SP) ;QUEUE DATA TO STACK.
1013 005000 062767 000002 174074 ADD #2,TYPQ2 ;UPDATE QUEUE POINTER.
1014 005006 105367 173775 DECB TYPQ2 ;DECREMENT COUNT.
1015 005012 000207 RTS PC ;EXIT, LEAVE DATA IN STACK.
    
```

```

1016 ;END CALL SERVICED HERE.
1017
1018 005014 011101 ENDSVC: MOV (R1),R1 ;GET START ADDR.
1019 005016 004767 177646 JSR PC,ENDCOM ;DO COMMON STUFF.
1020 005022 052761 020000 000020 ENDSVA: BIS #BIT13,STAT(R1) ;SET STOP BIT IN MODULE STAT.
1021 005030 105767 174317 TSTB CLOCK ;CLOCK AVAILABLE ?
1022 005034 001413 BEQ 1$ ;NO, CONTINUE
1023 005036 004767 013010 JSR PC,ERRTYM ;YES, SETUP TIME OF DROP
1024 005042 004567 002302 JSR R5,TYPEN ;TYPE
1025 005046 022744 CRCR ;2 CARRAGE RETURNS
1026 005050 026066 TOTIM ;TOTAL TIME
1027 005052 026111 PSTIM ;MODULE PASS TIME
1028 005054 025721 AEND ;COMMON HEADER
1029 005056 025775 MODEND ;MODULE DROPPED
1030 005060 000000 0 ;THAT'S ALL
1031 005062 000406 BR 2$ ;CONTINUE
1032
1033 005064 004567 002260 1$ JSR R5,TYPEN ;TYPE
1034 005070 022744 CRCR ;2 CARRAGE RETURNS
1035 005072 025721 AEND ;COMMON HEADER.
1036 005074 025775 MODEND ;END MESSAGE.
1037 005076 000000 0
1038 005100 105067 173710 2$ CLRBB TTYBSY ;CLEAR TTY BUSY INDICATOR.
1039 005104 005761 000020 ENDSVE: TST STAT(R1) ;BACKGROUND MODULE?
1040 005110 100410 BMI 1$ ;BR IF NOT.
1041 005112 105067 173675 CLRBB BRAKE ;RELEASE BRAKE.
1042 005116 105767 173702 TSTR STOP ;STOP MODULES FOR RELOCATION?
1043 005122 001403 BEQ 1$ ;NO, CONTINUE
1044 005124 105767 174224 TSTB ALLBK ;RUN ALL BKMODS BEFORE RELOCATION ?
1045 005130 001413 BEQ ENDSVG ;NO, CONTINUE
1046 005132 032761 010000 000020 1$ BIT #BIT12,STAT(R1) ;EXTENDED I/O MOD?
1047 005140 001402 BEQ 6$ ;NO- BRANCH
1048 005142 105367 173654 DECB MDXCNT ;YES, DEC EXTENDED MOD COUNT
1049 005146 105367 173643 6$ DECB MODCNT ;DECR COUNT OF MODULES RUNNING.
1050 005152 105767 173646 TSTB STOP ;RELOCATE CODE?
1051 005156 001445 BEQ ENDSVF ;NO- BRANCH
1052 005160 126767 173631 173637 ENDSVG: CMPB MODCNT,BKCNT ;ALL I/O MODULES OFF?
1053 005166 003046 BGT ENDSVD ;NO- BRANCH
1054 005170 105767 174160 TSTR ALLBK ;RUN ALL BKMOD'S BEFORE RELOCATING ?
1055 005174 001404 BEQ 1$ ;NO, CONTINUE
1056 005176 105767 173613 TSTB MDXCNT ;ALL MODULES DONE ?
1057 005202 001404 BEQ ENDSVB ;YES, GO RELOCATE
1058 005204 000437 BR ENDSVD ;NO, GO RUN THE REST OF THE MODULES
1059 005206 105767 173601 1$ TSTB BRAKE ;YES- BKMOD OFF?
1060 005212 001034 BNE ENDSVD ;NO- BRANCH (STOP BKMOD CLEANLY)
1061 005214 016767 174022 174022 ENDSVB: MOV OFFSET,OFFSTD ;YES- CALCULATE NEW OFFSET
1062 005222 066767 174074 174014 ADD ROTSI2,OFFSTD ;BY STEPPING AHEAD 8K OR 16K
1063 005230 026767 174010 174002 CMP OFFSTD,OFFMX ;NEW ADDRESS IN RANGE?
1064 005236 101402 RLOS 3$ ;YES- BRANCH
1065 005240 005067 174000 CLP OFFSTD ;NO- START OVER IN BANK 0
1066 005244 004767 013064 3$ JSR PC,CLKOFF ;MAKE SURF SYSTEM CLOCK IS OFF
1067 005250 004767 014220 JSR PC,MVCODE ;MOVE PROGRAM
1068 005254 004767 013136 JSR PC,CLKON ;TURN SYSTEM CLOCK BACK ON
1069 005260 112767 000001 174040 MOVB #1,RUNRFL ;SET FLAG TO INDICATE MODULE RESTART
1070 005266 000167 003642 JMP RUNRES ;GO STARTUP AGAIN
1071 005272 105767 173517 ENDSVF: TSTB MDXCNT ;ANY MODULES RUNNING?
    
```

```

1072 005276 001002          BNE      ENDSVD      ;YES= BRANCH
1073 005300 000167 001544    JMP      CTRLCB     ;COUNT 0, TERMINATE RUN MODE.
1074 005304 000167 174704    ENDSVD; JMP      QUETST ;GO BACK TO SERVICE QUEUES.
    
```

```

1075          ;PASEND ROUTINE, TYPES END OF PASS MESSAGE.
1076          ;BACKGROUND MODULES ARE NOT ALLOWED TO MAKE MULTIPLE PASSES.
1077
1078 005310 011101          PASEND; MOVB      (R1),R1      ;LOAD R1 WITH MODULE ADDRESS
1079 005312 016167 000062 174172 MOVB      RSTR1(R1),CADDR    ;SAVE RESTART ADDRESS
1080 005320 010167 174170 MOVB      R1,CSTART        ;SAVE MODULE ADDRESS
1081 005324 005261 000026 MOVB      PSCNT(R1)        ;INCREMENT PASS COUNT.
1082 005330 105767 174017 TSTB     CLOCK            ;IS THERE A CLOCK AVAILABLE?
1083 005334 001404 BEQ      1#              ;NO, CONTINUE
1084 005336 004767 013120 JSR      PC,CKHUNG        ;GO CHECK FOR ANY "PUNG" MODULES
1085 005342 004767 012514 JSR      PC,PASTYM        ;SET UP RUNTIME, PASTIME INFORMATION
1086 005346 032777 010000 173622 1#; BIT      #BIT12,@SR        ;ENDPAS PRINTOUT?
1087 005354 001432 BEQ      PSEND          ;BR IF NOT.
1088 005356 004767 177306 JSR      PC,ENDCOM        ;DO COMMON STUFF.
1089 005362 016146 000026 MOVB      PSCNT(R1),-(SP) ;NOW GET IT.
1090 005366 004567 007000 JSR      RS,BDCNV        ;CONVERT IT TO DECIMAL ASCII.
1091 005372 026016 BPSCNT   CLOCK            ;STUFF IT AT RPSCNT.
1092 005374 105767 173753 TSTB     CLOCK            ;IS THERE A CLOCK AVAILABLE?
1093 005400 001010 BNE      2#              ;YES, CONTINUE
1094 005402 004567 001742 JSR      RS,TYPEN        ;TYPE
1095 005406 022744 CRCP     ;2 CARRAGE RETURNS
1096 005410 025721 AEND     ;COMMON HEADER.
1097 005412 026007 APSEND   ;ENDPAS AND COUNT.
1098 005414 022742 CR      ;ENDPAS AND COUNT.
1099 005416 000000 CR      0
1100 005420 000410 BR      PSEND          ;CONTINUE
1101 005422 004567 001722 2#; JSR      RS,TYPEN        ;TYPE
1102 005426 022744 CRCP     ;2 CARRAGE RETURNS
1103 005430 025721 AEND     ;COMMON HEADER
1104 005432 026007 APSEND   ;END PAS AND COUNT
1105 005434 026066 TOTIM    ;TOTAL TIME
1106 005436 026111 PSTIM    ;PASS TIME
1107 005440 000000 CR      0
1108 005442 105067 173346 PSEND; CLR      TTYBSY     ;THAT'S ALL
1109 005446 105767 173640 TSTB     CHN            ;CLEAR TTY BUSY INDICATOR.
1110 005452 100032 BPL      10#           ;IN CHAIN MODE?
1111          ;BR IF NOT
1112          ;ELSE MAKE SURE EVERY MODULE KEEPS RUNNING TILL ALL HAVE COMPLETED
1113          ;AN END OF PASS, THEN STOP ONE AT A TIME, THEN RETURN TO CHAIN MONITOR
1114 005454 010046 MOVB      RO,-(SP)        ;SAVE R0
1115 005456 010146 MOVB      R1,-(SP)        ;SAVE R1
1116 005460 012700 026520 MOVB      #MODQ-2,R0      ;GET LIST OF MODULE ADDRESS
1117 005470 011001 11#; ADD      #2,R0          ;LOOK AT NEXT ENTRY
1118 005472 001416 MOVB      (R0),R1        ;ANY MODULES LEFT?
1119 005474 032761 040000 000020 BIT      12#            ;BR IF NOT, ALL DONE, DO NOT RESTART MODULES
1120 005502 001770 BEQ      #BIT14,STAT(R1) ;MODULE SELECTED?
1121 005504 032761 020000 000020 BIT      11#            ;BR IF NOT
1122 005512 001364 BNE      #BIT13,STAT(R1) ;WAS IT DROPPED?
1123 005514 005761 000026 TSTB     11#            ;BR IF YES
1124 005520 001361 BNE      PSCNT(R1)      ;HAS HE DONE A PASS YET?
1125 005522 012601 MOVB      11#            ;BR IF YES
1126 005524 012600 MOVB      (SP)+,R1      ;RESTORE R
1127 005526 000404 MOVB      (SP)+,R0      ;RESTORE R0
1128 005530 012601 BR      10#            ;GO LET THIS GUY DOING THE EOP RUN AGAIN
1129 005532 012600 MOVB      (SP)+,R1      ;RESTORE R1
1130 005534 000167 177344 MOVB      (SP)+,R0      ;RESTORE R0
1131          JMP      ENDSVD ;START LETTING MODULES DIE OFF
    
```

```

1131 005540 105767 173260      100:  TSTB  STOP          ;STOP ALL MODULES TO ROTATE CODE?
1132 005544 001402              BEQ    140          ;NO BR.
1133 005546 000167 177332      JMP    ENDSVE
1134 005552 105267 173254      140:  INCB  PASTOT      ;COUNT TOTAL PASSES AT PRESENT OFFSET
1135 005556 105767 173531      TSTB  ROTI          ;WRITE BUFFER ROTATION ENABLED?
1136 005562 001403              BEQ    10          ;NO, TREAT LIKE ALL BKMDS RUNNING
1137 005564 105767 173232      TSTB  MDXCNT       ;ANY EXTENDED MODULES RUNNING?
1138 005570 001021              BNE   20          ;YES- SKIP ALTERNATE RELOCATION METHOD
1139 005572 126767 173234 173233 10:  CMPB  PASTOT,PASREL ;NO- UP TO RELOCATION NUMBER?
1140 005600 103415              BLO   20          ;NO- BRANCH
1141 005602 105767 173507      TSTB  RELOC        ;YES- RELOCATION ALLOWED?
1142 005606 001412              BEQ    20          ;NO- BRANCH
1143 005610 105767 173513      TSTB  LOCK        ;LOCK SET TO INHIBIT RELOCATION?
1144 005614 001007              BNE   20          ;YES- BRANCH
1145 005616 105767 173530      TSTB  WSTOP       ;ALLOW STOP TO SET ?
1146 005622 001004              BNE   20          ;NO, CONTINUE
1147 005624 105267 173174      INCB  STOP        ;YES, SET STOP
1148 005630 000167 177250      JMP    ENDSVE
1149 005634 005761 000020      20:  TST   STAT(R1)    ;BACKGROUND MODULE?
1150 005640 100415              BMI   PSEENB      ;BR IF NOT.
1151 005642 105067 173145      CLRB  BRAKE       ;RELEASE BRAKE.
1152 005646 032761 001000 000020  BIT   #BIT9,STAT(R1) ;IS IT AN NBKMOD ?
1153 005654 001613              BEQ   ENDSVD      ;NO, CONTINUE
1154 005656 052761 020000 000020  BIS   #BIT13,STAT(R1) ;YES, SET IT'S STOP BIT
1155 005664 105367 173135      DECB  SKCNT       ;ONE LESS BKMDS RUNNING
1156 005670 000167 177210      JMP   ENDSVE     ;GO TAKE CARE OF OTHER STUFF
1157 005674 000167 173602      PSEENB; JMP    COMQUE ;GO TO COMMON QUE CALL.
1158
1159
    
```

```

1160                                     ;TRACE TRAP ENTERS HERE.
1161
1162 005700 105367 173130      TRCI:  DECB  BKTMR
1163 005704 100401              BMI   TRCIC
1164 005706 000006              TRCIA:  RTT
1165 005710 116767 173422 173116  TRCIC:  MOVB  BKTMR,BKTMR
1166 005716 005767 173064      TST   IOQUE      ;I/O OR TYPE QUE WAITING?
1167 005722 001771              BEQ   TRCIA      ;NO, EXIT.
1168 005724 012667 173140      TRCIB:  MOV   (SP)+,TRCPC ;SAVE MOD'S PC.
1169 005730 012667 173136      MOV   (SP)+,TRCPSW ;SAVE MOD'S PSW.
1170 005734 000401              BR    EXIT1.
    
```

```

1171                                     ;EXIT CALL ENTERS HERE.
1172
1173 005736 022626                       EXIT.1  CMP      (SP)+,(SP)+
1174 005740 010046                       EXIT1.1  MOV      RO,-(SP)
1175 005742 016700 173170                MODADR,RO ;SAVE RO IN STACK,
1176 005746 062700 000032                ADD      #SVRO,RO ;MODULE ADDR TO RO.
1177 005752 012620                        MOV      (SP)+,(RO)+ ;POINT TO MOD'S REG SAVE AREA.
1178 005754 010120                        MOV      R1,(RO)+ ;SAVE RO, (FROM STACK).
1179 005756 010220                        MOV      R2,(RO)+ ;SAVE REMAINING REGS.
1180 005760 010320                        MOV      R3,(RO)+
1181 005762 010420                        MOV      R4,(RO)+
1182 005764 010520                        MOV      R5,(RO)+
1183 005766 010610                        MOV      R6,(RO)
1184 005770 016706 173156                SPSAV,R6 ;SAVE MODULE STACK POINTER.
1185 005774 005046                       EXIT2.1 CLR      =(SP) ;RESTORE MONITOR STACK.
1186 005776 012746 006004                MOV      #16,=(SP) ;CLEAR PSW ON STACK
1187 006002 000002                        RTI      ;SET UP RETURN PC
1188 006004 000167 174204                18:     JMP      QUETST ;CLEAR PSW AND CONTINUE
    
```

```

1189                                     ;TYP04. ROUTINE, LOADS MSGD6 INFORMATION INTO TYPEQ
1190
1191 006010 004767 000242                TYP04.1 JSR      PC,LDTYPEQ ;QUEUE PC+2 OF CALL
1192 006014 016700 173116                MOV      MODADR,RO ;LOAD MODULE ADDRESS IN RO
1193 006020 010046                        MOV      RO,-(SP) ;AND SAVE IN QUEUE
1194 006022 004767 000230                JSR      PC,LDTYPEQ
1195 006026 016046 000060                MOV      #HAS(RO),=(SP) ;QUEUE DATA ERROR COUNT
1196 006032 004767 000220                JSR      PC,LDTYPEQ
1197 006036 016046 000056                MOV      ASB(RO),=(SP) ;QUEUE SIZE OF TRANSFER
1198 006042 004767 000210                JSR      PC,LDTYPEQ
1199 006046 005726                        POPSP
1200 006050 016706 173076                MOV      SPSAV,R4 ;REMOVE PS FROM STACK
1201 006054 000747                        BR       EXIT2. ;GET MONITOR STACK POINTER
    
```

1202  
 1203  
 1204 006056 011846  
 1205 006060 004767 000172  
 1206 006064 016700 173046  
 1207 006070 010046  
 1208 006072 004767 000160  
 1209 006076 017616 000000  
 1210 006102 004767 000150  
 1211 006106 005726  
 1212 006110 062700 000062  
 1213 006114 012701 000005  
 1214 006120 000401  
 1215 006122 005010  
 1216 006124 014046  
 1217 006126 004767 000124  
 1218 006132 005301  
 1219 006134 001372  
 1220 006136 016700 172774  
 1221 006142 010660 000046  
 1222 006146 016708 173000  
 1223 006152 000167 177616

;TYPQ3. ROUTINE QUEUES INFORMATION FOR DERRX CALLS

TYPQ3: MOV (SP),-(SP)  
 JSR PC,LDTPQ ;QUEUE PC+2 OF CALL  
 MOV MODADR,R0 ;LOAD MODULE ADDRESS IN R0  
 MOV R0,=(SP) ;AND SAVE IN QUEUE  
 JSR PC,LDTPQ  
 MOV R0,=(SP) ;QUEUE VIRTUAL ADDRESS OF CHECKDATA CALL  
 JSR PC,LDTPQ  
 POPSP  
 ADD #ANAS+2,R0 ;GET ADDRESS OF INFORMATION  
 MOV #5,R1  
 BR 28  
 18: CLR (R0) ;CLEAR DATA (EXCEPT LAST)  
 28: MOV =(R0),=(SP) ;GET DATA  
 JSR PC,LDTPQ ;QUEUE DATA  
 DEC R1  
 BNE 18  
 MOV MODADR,R0  
 MOV R6,SVR6(R0)  
 MOV SPSAV,R6  
 JMP EXIT2.

1224  
 1225  
 1226 006156 011646  
 1227 006160 004767 000072  
 1228 006164 010046  
 1229 006166 017600 000002  
 1230 006172 062700 000050  
 1231 006176 012746 000005  
 1232 006202 012046  
 1233 006204 000402  
 1234 006206 011046  
 1235 006210 005020  
 1236 006212 004767 000040  
 1237 006216 005316  
 1238 006220 001372  
 1239 006222 005726  
 1240 006224 012600  
 1241 006226 000167 177504

;TYPQ2. ROUTINE SERVICES ERROR, DATA ERROR, AND ERRN CALLS.

TYPQ2: MOV (SP),-(SP) ;SAVE PC+2 OF CALL AGAIN.  
 JSR PC,LDTPQ ;QUEUE UP PC+2 OF CALL.  
 MOV R0,=(SP) ;SAVE R0.  
 MOV R2,(SP),R0 ;GET MODULE ADDR.  
 TYPQX: ADD #CSRA,R0  
 MOV #5,=(SP)  
 MOV (R0)+,=(SP)  
 BR 28  
 18: MOV (R0),-(SP)  
 CLR (R0)+  
 28: JSR PC,LDTPQ  
 DEC (SP)  
 BNE 18  
 POPSP  
 MOV (SP)+,R0 ;RESTORE R0.  
 JMP EXIT.



```

1252 ;ROUTINE TO LOAD TYPE QUEUE, ENTERED VIA JSR PC,
1253 LDTYPEQ; CLR =(SP) ;CLEAR PSW ON STACK
1254 006256 005046 006266 MOV #3,-(SP) ;SETUP RETURN PC
1255 006264 000002 RTI ;CLEAR PSW AND CONTINUE
1256 006266 105767 172515 3: TSTB TYPQUE ;REQUEST COUNT 0?
1257 006272 001414 BEQ 1: ;BR IF YES.
1258 006274 026767 172600 172600 CMP TYPQ1,TYPQ2 ;NO, TYPQ1 AND TYPQ2 SAME?
1259 006302 001010 BNE 1: ;BR IF NOT.
1260 006304 012767 030072 172566 MOV #TYPEQ,TYPQ1 ;RESET THE TYPE QUEUE
1261 006312 104406 022774 MSG$,GOVRFL ;YES, QUEUE OFLO, CRASH SYSTEM
1262 006316 104406 023746 MSG$,HLT ;'HALT'
1263 006322 000000 HALT
1264 006324 026727 172550 031222 1: CMP TYPQ1,#TYPLIM ;REACHED HIGH LIMIT?
1265 006332 001003 BNE 2: ;BR IF NOT.
1266 006334 012767 030072 172536 MOV #TYPEQ,TYPQ1 ;RESET TYPQ1.
1267 006342 016677 000002 172530 2: MOV 2(SP),#TYPQ1 ;STORE ARGUMENT IN QUEUE
1268 006350 105267 172433 INCB TYPQUE ;UPDATE REQUEST COUNTS.
1269 006354 062767 000002 172516 ADD #2,TYPQ1 ;UPDATE TYPQ1.
1270 006362 012616 MOV (SP)+,(SP) ;
1271 006364 000207 RTS PC ;EXIT.
    
```

```

1272 ;ROUTINE TO LOAD BREAK$ CALL
1273
1274 006366 112766 177777 000003 LDBRK; MOVB #1,3(SP) ;INDICATE BREAK$ CALL
1275 006374 000402 BR ;
1276
1277 ;ROUTINE TO LOAD QUES$ CALL.
1278
1279 006376 005066 000002 QUE; CLR 2(SP) ;INDICATE QUE CALL.
1280
1281 ;LDIOQ ROUTINE. MUST NOT DROP BELOW LEVEL OF INTERRUPTING DEVICE.
1282
1283 006402 012746 000340 LDIOQ; MOV #PRTY7,-(SP) ;PUT NEW PSW ON STACK
1284 006406 012746 006414 MOV #1,-(SP) ;SETUP RETURN PC
1285 006412 000002 RTI ;LOAD NEW PSW AND CONTINUE
1286 006414 000403 1: BR ,+10
1287 006416 112766 000001 000002 PIRQ; MOVB #1,2(SP) ;REQUEST COUNT 0?
1288 006424 105767 172356 TSTB IOQUE ;BR IF YES.
1289 006430 001414 BEQ 1: ;IOQ1 AND IOQ2 SAME?
1290 006432 026767 172436 172436 CMP IOQ1,IOQ2 ;BR IF NOT.
1291 006440 001010 BNE 1: ;RESET IOQ1
1292 006442 012767 027252 172424 MOV #IOQ,IOQ1 ;QUE OFLO, CRASH SYSTEM
1293 006450 104406 022774 MSG$,GOVRFL ;'HALT'
1294 006454 104406 023746 MSG$,HLT
1295 006460 000000 HALT
1296 006462 026727 172406 030072 1: CMP IOQ1,#IOQLIM ;REACHED HIGH LIMIT?
1297 006470 001003 BNE 2: ;BR IF NOT.
1298 006472 012767 027252 172374 2: MOV #IOQ,IOQ1 ;RESET IOQ1.
1299 006500 012677 172370 2: MOV (SP)+,#IOQ1 ;STORE PC+2 OF PENDING CALL.
1300 006504 105267 172276 INCB IOQUE ;UPDATE REQUEST COUNTS.
1301 006510 062767 000002 172356 ADD #2,IOQ1 ;UPDATE IOQ1.
1302 006516 005716 TST (SP) ;QUE CALL?
1303 006520 001003 BNE 3: ;NO= BRANCH
1304 006522 005726 TST (SP)+ ;YES, POP STACK
1305 006524 000167 177244 JMP EXIT2. ;EXIT
1306 006530 126627 000001 177777 3: CMPB 1(SP),#-1 ;BREAK CALL?
1307 006536 001003 BNE 4: ;NO= BRANCH
1308 006540 005726 TST (SP)+ ;YES= POP STACK
1309 006542 000167 177172 JMP EXIT1. ;
1310 006546 005726 4: TST (SP)+ ;PIRQ= EXIT VIA RTI
1311 006550 000002 RTI
    
```

```

1312                                     ;KEYBOARD INTERRUPT SERVICE
1313
1314 006552 005077 172404 KBSRVC CLR @TKS ;CLEAR INTERRUPT ENABLE.
1315 006556 017746 172402 MOV @TKB,-(SP) ;GET CHAR.
1316 006562 042716 000200 BIC #200,(SP) ;REMOVE EXTRA BITS.
1317 006566 122716 000003 CMPB #3,(SP) ;IS IT CTRL C?
1318 006572 001524 BEQ CTRLCA ;BR IF YES.
1319 006574 005726 POPSP ;REMOVE CHAR FROM STACK.
1320 006576 000004 006604 000000 PIRQ#18,0 ;Q DEVICE SERVICE.
1321
1322                                     ;KEYBOARD SERVICE.
1323
1324 006604 026727 172226 026752 18: CMP KBPTR,#KBUF+KBÜFL ;BUFFER EXCEEDED?
1325 006612 001511 BEQ 128 ;BR IF YES.
1326 006614 017746 172344 MOV @TKB,-(SP) ;GET CHAR.
1327 006620 042716 000200 BIC #200,(SP) ;CLEAR PARITY BIT.
1328 006624 121627 000177 CMPB (SP),#177 ;RUBOUT?
1329 006630 001443 BEQ 58 ;BR IF YES.
1330 006632 121627 000040 CMPB (SP),#40 ;SPACE? (IGNORE LEADING ONES).
1331 006636 001460 BEQ 88 ;BR IF YES.
1332 006640 121627 000015 CMPB (SP),#15 ;CR? (TIME TO QUIT?).
1333 006644 001462 BEQ 98 ;BR IF YES.
1334 006646 121627 000012 CMPB (SP),#12 ;LF? (TREAT LIKE CR).
1335 006652 001457 BEQ 98 ;BR IF YES.
1336 006654 121627 000060 CMPB (SP),#60 ;CHAR LESS THAN 60?
1337 006660 103425 BLO 108 ;BR IF YES, IGNORE IT.
1338 006662 121627 000141 CMPB (SP),#141 ;CHAR LESS THAN LOWER CASE A?
1339 006666 103405 BLO 28 ;BR IF YES.
1340 006670 121627 000172 CMPB (SP),#172 ;CHAR HIGHER THAN LOWER CASE Z?
1341 006674 101017 BHI 108 ;BR IF YES.
1342 006676 162716 000040 SUB #40,(SP) ;LOWER CASE CHAR, MAKE IT UPPER CASE.
1343 006702 111677 172130 MOVB (SP),#KBPTR ;STORE CHAR.
1344 006706 005267 172124 INC KBPTR ;UPDATE BUFFER POINTER.
1345 006712 112667 172342 MOVB (SP)+,TTYBYT ;CHAR TO ECHO BUFFER.
1346 006716 104406 001260 MSGS,TTYBYT ;ECHO THE CHARACTER.
1347 006722 052777 000100 BIS #IE,@TKS ;REENABLE KYBD INTERRUPTS.
1348 006730 000167 173260 JMP QUIETST
1349 006734 005726 108: POPSP
1350 006736 000771 BR 48
1351
1352                                     ;SERVICE DELETE/RUBOUT CODE.
1353
1354 006740 022767 026712 172070 58: CMP #KBUF,KBPTR ;BUFFER EMPTY?
1355 006746 001405 BEQ 68 ;BR IF YES.
1356 006750 005367 172062 DEC KBPTR ;POINT TO PREVIOUS CHAR.
1357 006754 117716 172056 MOVB #KBPTR,(SP) ;GET PREVIOUS CHAR.
1358 006760 000754 BR 38 ;GO ECHO IT.
1359 006762 005726 68: POPSP
1360 006764 104406 022772 MSGS,ACRLF ;OUTPUT CRLF.
1361 006770 012767 026712 172040 78: MOV #KBUF,KBPTR ;RESET KBUF POINTER.
1362 006776 000751 BR 48
1363
1364                                     ;SERVICE SPACE.
1365
1366 007000 022767 026712 172030 88: CMP #KBUF,KBPTR ;LEADING SPACE?
1367 007006 001741 BEQ 38 ;BR IF YES (IGNORE IT).
    
```

```

1368 007010 000734 BR 28 ;STORE IT.
1369
1370                                     ;SERVICE CR AND LF.
1371
1372 007012 112677 172020 98: MOVB (SP)+,#KBPTR ;STORE CHAR.
1373 007016 012767 026712 172012 MOV #KBUF,KBPTR ;POINT TO START OF KBUF.
1374 007024 104406 022772 MSGS,ACRLF ;OUTPUT CRLF.
1375 007030 104401 010512 000000 QUES,DECODE,0 ;QUE TO SERVICE COMMAND.
1376
1377                                     ;SERVICE KBUF OVERFLOW.
1378
1379 007036 104406 023505 128: MSGS,KBÜFLO ;TYPE KBUF OFLO.
1380 007042 000752 BR 78
    
```

```

1381 ;SERVICE CTRL C, ENDS RUN MODE ALSO,
1382 ;DROPS TO BANKS 0 AND 1 BEFORE RETURNING (IF KT11 IN USE)
1383 ;STACK POINTER SHOULD BE SETUP AHEAD OF CALL TO CTRLX TO PRESERVE
1384 ;NEEDED INFORMATION PRIOR TO REMAPPING TO 0, OR CAUSE REMAPPING TO
1385 ;CURRENT AREA AFTER
1386
1387 007044 012706 027212 CTRLCA; MOV #SPBOT,SP ;REINITIALIZE STACK POINTER
1388 007050 004767 000130 CTRLCB; JSR PC,CTRLX ;CLEAR QUEUES, TYPE "C
1389 007054 005067 172164 CLR OFFSTD ;GET PROGRAM BACK TO BANK 0
1390 007060 004767 012410 JSR PC,MVCODE
1391 007064 016767 012054 172074 MOV LPWK1,TPS ;RESTORE TTY
1392 007072 016767 012050 172070 MOV LPWK2,TPB
1393 007100 105767 172205 TSTB RMODE ;IN RUN MODE?
1394 007104 001002 BNE 18 ;BR IF YES,
1395 007106 000167 001544 JMP COMCO3 ;BACK TO KYBD ROUTINE,
1396 007112 105067 172173 18: CLRB RMODE ;CLEAR RUN MODE INDICATOR
1397 007116 105767 172170 TSTB CHN ;IN CHAIN MODE?
1398 007122 100426 BMI CTRPLCD ;BR IF YES, BYPASS SUMMARY AND RETURN TO
1399 ;CHAIN MONITOR
1400 007124 052777 000100 172030 BIS #IE,ATKS ;REENABLE KEYBOARD INTERRUPTS
1401 007132 104406 022753 MSG8,SUMARY ;TYPE RUN END SUMMARY TITLE,
1402 007136 105767 172211 TSTB CLOCK ;IS THERE A CLOCK AVAILABLE?
1403 007142 001406 BEG 28 ;NO, CONTINUE
1404 007144 004767 010506 JSR PC,RUNTYM ;YES, SET UP RUNTIME INFORMATION
1405 007150 104406 022747 MSG8,AT ;"AT"
1406 007154 104406 026066 MSG8,TOTIM ;PRINT THE TOTAL TIME
1407 007160 104406 022742 28: MSG8,CR ;CARRIAGE RETURN
1408 007164 004767 003144 JSR PC,DIRA ;TYPE RUN SUMMARY,
1409 007170 004767 007744 JSR PC,SYSNUM ;TYPE NUMBER OF SYSTEM ERRORS
1410 007174 004767 010024 JSR PC,PWRNUM ;TYPE NUMREP OF POWER FAILS
1411 007200 000167 006700 CTRLCD; JMP CHNOUT ;EXIT, OR RETURN TO KYBD RTN,
1412
    
```

```

1413 007204 012746 000340 CTRLX; MOV #PRTY7,-(SP) ;SETUP NEW PSW ON STACK
1414 007210 012746 007216 MOV #48,-(SP) ;PUT RETURN PC ON STACK
1415 007214 000002 RTI ;LOAD NEW PSW AND CONTINUE
1416 007216 012767 000144 172270 48: MOV #100,,CSTART ;CLEAR QUEUES AND DELAY TOO,
1417
1418 007224 105767 172061 TSTB RMODE ;IF NOT IN RMODE, SHORTEN WAIT
1419 007230 001003 BNE 18
1420 007232 012767 000010 172254 MOV #10,CSTART
1421 007240 004767 005522 18: JSR PC,CLRQUS ;CLEAR QUEUES,
1422 007244 005367 172244 DEC CSTART ;DONE?
1423 007250 001373 BNE 18 ;BR IF NOT,
1424 007252 011604 MOV (SP),R4 ;SAVE RETURN PC
1425 007254 000005 RESET
1426 007256 010416 MOV R4,(SP) ;RESTORE RETURN PC IN CURRENT BANK
1427 007260 004767 005502 JSR PC,CLRQUS ;CLEAR QUEUES IN BANK 0
1428 007264 105767 172014 TSTB KTPRES
1429 007270 001405 BEG 28
1430 007272 004767 002516 JSR PC,KTINIT
1431 007276 012737 000001 177572 MOV #1,##SSPO
1432 007304 132767 000001 172026 28: BITB #R10,PPRES ;PARITY PRESENT ?
1433 007312 001402 RPO 38 ;NO, CONTINUE
1434 007314 004767 002712 JSR PC,PAENBL ;YES, GO TURN ON PARITY
1435 007320 005046 CLR -(SP) ;RETYPE NEW PSW ON STACK
1436 007322 012746 007330 MOV #58,-(SP) ;PUT RETURN PC ON STACK
1437 007326 000002 RTI ;LOAD NEW PSW AND CONTINUE
1438 007330 105267 171457 58: INCB BRAKE ;SET BRAKE TO PREVENT BACKGROUND MODULE FROM BEING START
1439 007334 016767 171760 MOV ROTNUM,ROTCNT ;RESET THE ROTATION COUNTER
1440 007342 104406 022770 MSG8,CTRPLC ;OUTPUT "C
1441 007346 000207 RTS PC ;EXIT,
    
```

```

1442                                     ;SUBROUTINE TO TYPE MULTIPLE ASCII STRINGS, NOTE THAT THE PREVIOUS
1443                                     ;VALUE OF R5 IS NOT SAVED,
1444
1445 007350 005726                         TYPE: TST      (SP)+          ;REMOVE OLD PS FROM STACK
1446 007352 012546                         TYPE: MOV      (5)+,-(SP)      ;GET ASCII ADDRESS,
1447 007354 001403                         BEQ          18              ;BR IF 0 TERMINATOR,
1448 007356 004767 000004                 JSR          PC,TYPE        ;TYPE IT,
1449 007362 000773                         BR           TYPE1         ;GET MORE,
1450 007364 000205                         18: RTS          R5         ;RETURN,
    
```

```

1451                                     ;TYPE SUBROUTINE,
1452                                     ;TYPES ASCII STRING
1453                                     ;NOTE THAT IF RMODE IS SET BUT MODULES AREN'T YET BEING STARTED, BRAKE
1454                                     ;MUST BE SET TO PREVENT STARTING MODULES WHILE TYPING
1455
1456 007366 012767 007436 172134 TYPE: MOV      #TTSRV1,TTYLNK
1457 007374 012667 171506                 MOV      (SP)+,TYPRET      ;SAVE RETURN ADDRESS
1458 007400 012667 171500                 MOV      (SP)+,TYPADR      ;SAVE MESSAGE ADDRESS
1459 007404 010167 171530                 MOV      R1,TYPR1         ;SAVE REGISTERS 1-5
1460 007410 012701 001142                 MOV      #TYPR2,R1
1461 007414 010221                 MOV      R2,(R1)+
1462 007416 010321                 MOV      R3,(R1)+
1463 007420 010421                 MOV      R4,(R1)+
1464 007422 010521                 MOV      R5,(R1)+
1465 007424 012777 000100 171534         MOV      #100,#TPS        ;SET IE
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486 007432 000167 172556                 TTSRV: JMP      QUETST
1487 007436 117746 171442                 MOV      #TYPADR,-(SP)    ;GET CHAR,
1488 007442 001013                         BNE      TYPEB           ;BR IF NOT TERMINATOR,
1489 007444 005726                         TST      (SP)+
1490 007446 012701 001142                 MOV      #TYPR2,R1      ;RESTORE REGISTERS 1-5
1491 007452 012102                 MOV      (R1)+,R2
1492 007454 012103                 MOV      (R1)+,R3
1493 007456 012104                 MOV      (R1)+,R4
1494 007460 012105                 MOV      (R1)+,R5
1495 007462 016701 171452                 MOV      TYPR1,R1
1496 007466 005077 171474                 CLR      #TPS           ;CLEAR IE
1497 007472 000177 171410                 JMP      #TYPRET
1498
1499
1500 007476 062767 000001 171400 TYPE: ADD      #1,TYPADR
1501 007504 122716 000045                 CMPB     #45,(SP)        ;IS IT ?
1502 007510 001037                         BNE      TYPEL           ;NO- BRANCH
1503 007512 112716 000015 TYPE: MOV      #15,(SP)    ;OUTPUT CR,
1504 007516 012767 007526 172004         MOV      #16,TTYLNK
1505 007524 000431                         BR       TYPEL
1506 007526 112746 000012 18: MOV      #12,-(SP)    ;OUTPUT LF,
    
```

```

1507
1508
1509
1510 007532 012767 007542 171770      MOV      #28,TTYLNK
1511 007540 000423                    BR       TYPEL
1512 007542 116767 171540 171250 28i  MOVVB   FILCNT,FILCTR ;GET FILL COUNT.
1513 007550 001002                    BNE     TYPEE ;BR IF NOT 0.
1514 007552 105267 171242                    INCB   FILCTR ;OOPS, MAKE IT A 1.
1515 007556 116746 171525                    TYPEE; MOVVB  FILLER,-(SP) ;OUTPUT FILLER.
1516 007562 012767 007572 171740 TYPEE: MOV      #38,TTYLNK
1517 007570 000407                    BR       TYPEL
1518 007572 105367 171222                    38i    DECB   FILCTR ;DECREMENT FILL COUNTER.
1519 007576 001367                    BNE     TYPEE ;BR IF NOT 0.
1520 007600 012767 007436 171722                    MOV      #TTSRV1,TTYLNK
1521 007606 000713                    BR       TTSRV1
1522 007610 012677 171354                    TYPEL: MOV   (SP)+,@TPB
1523 007614 012767 010000 171266                    MOV      #10000,TYPCTR
1524 007622 000167 172366                    JMP      QUETST
    
```

```

1525 ;ROUTINE TO SERVICE CKDATA CALL (CDATA8)
1526
1527 007626 112767 177777 171513 DATCK,i MOVB   #+1,DATFLG ;SET FLAG FOR DATACK CALL
1528 007634 000402                    BR       CKCONT ;CONTINUE
1529 007636 105067 171505                    CDATA,i CLRB  DATFLG ;SET FLAG FOR CKDATA CALL
1530
1531 007642 005067 171310                    CKCONT: CLR   CDPLG
1532 007646 005067 171256                    CLR   DERRCT ;INITIALIZE DATA ERROR COUNT
1533 007652 004467 005062                    JSR   R4,SAV04 ;TEMPORARILY SAVE REGS
1534 007656 016604 000012                    MOV   12(SP),R4 ;GET PC+2 OF CALL
1535 007662 010467 171632                    CDATA: MOV   R4,CDERR1 ;SETUP PC OF CALL
1536 007666 162767 000002 171624                    SUB   #2,CDERR1
1537 007674 012401                    MOV   (R4)+,R1 ;GET BEGIN ADDRESS OF MODULE
1538 007676 012403                    MOV   (R4)+,R3 ;GET TABLE ADDRESS OF RBUPPA FROM MODULE
1539 007700 010361 000052                    MOV   R3,SBADR(R1) ;SAVE IT
1540 007704 062701 000032                    ADD   #SVRO,R1 ;SAVE MODULE'S REGISTERS
1541 007710 012621                    MOV   (SP)+,(R1)+
1542 007712 012621                    MOV   (SP)+,(R1)+
1543 007714 012621                    MOV   (SP)+,(R1)+
1544 007716 012621                    MOV   (SP)+,(R1)+
1545 007720 012621                    MOV   (SP)+,(R1)+
1546 007722 010511                    MOV   R5,(R1)
1547 007724 022626                    CMP   (SP)+,(SP)+
1548 007726 016401 177774                    MOV   -(R4),R1 ;RESTORE MODULE ADDR
1549 007732 012300                    MOV   (3)+,R0 ;GET READ BUFFER PA
1550 007734 012367 171164                    MOV   (3)+,MODREA ;GET EA BITS ALSO
1551 007740 012305                    MOV   (3)+,R5 ;BUFFER WORD COUNT
1552 007742 016102 000074                    MOV   WBUFFA(R1),R2 ;GET WRITE BUFFER PA
1553 007746 016167 000076 171142                    MOV   WBUFEA(R1),MODWEA ;AND GET EA BITS
1554 007754 005003                    CLR   R3 ;CLEAR OUT REG. 3
1555 007756 005767 171174                    TST   CDPLG ;ENTERED FROM DATA ERROR RETURN?
1556 007762 001421                    BEQ   308 ;NO- BRANCH
1557 007764 012603                    MOV   (SP)+,R3 ;YES- GET COUNT
1558 007766 005203                    INC   R3
1559 007770 006303                    ASL   R3 ;CREATE OFFSET
1560 007772 050300                    ADD   R3,R0 ;GET ADDRESS OF CURRENT READ BUFFER WORD
1561 007774 103003                    RCC   #+10
1562 007776 062767 000020 171120                    ADD   #20,MODREA
1563 010004 060302                    ADD   R3,R2 ;GET ADDRESS OF CURRENT WRITE BUFFER WORD
1564 010006 103003                    RCC   #+10
1565 010010 062767 000020 171100                    ADD   #20,MODWEA
1566 010016 006203                    ASR   R3 ;RESTORE COUNT
1567 010020 160305                    SUB   R3,R5 ;CALCULATE WORDS LEFT
1568 010022 012667 171102                    MOV   (SP)+,DERRCT ;RESTORE DATA ERROR COUNT
1569 010026 010067 171070                    308:  MOV   R0,MODRPA
1570 010032 010267 171056                    MOV   R2,MODWPA
1571 010036 004567 004566                    JSR   R5,TOPI26 ;GET 12 BIT WRITE BUFFER ADDRESS
1572 010042 001114                    MODWPA
1573 010044 004567 004552                    JSR   R5,TOPI26 ;GET 12 BIT READ BUFFER PA
1574 010050 001122                    MODRPA
1575 010052 105767 171226                    TSTB  KTPRES ;USE KT11?
1576 010056 001443                    BEQ   18 ;NO- BRANCH
1577 010060 012767 077406 167512                    MOV   #77406,UIPDR0 ;MAP USER R/W
1578 010066 012767 077406 167506                    MOV   #77406,UIPDR1
1579 010074 012767 077406 167502                    MOV   #77406,UIPDR2
1580 010102 012767 077406 167506                    MOV   #77406,UIPDR7
    
```

1581	010110	016767	162224	167522		MOV	KIPARO,UIPARO	;EXECUTE IN PAGE 0
1582	010116	016767	170776	167516		MOV	MODW12,UIPAR1	;PAGE 1 IS WRITE BUFFER
1583	010124	016767	170776	167512		MOV	MODR12,UIPAR2	;PAGE 2 IS READ BUFFER
1584	010132	012767	007600	167516		MOV	#7600,UIPAR7	;PAGE 7 IS EXTERNAL
1585	010140	042700	177700			BIC	#177700,R0	;SETUP READ BUFFER VA
1586	010144	052700	040000			BIS	#40000,R0	
1587	010150	042702	177700			BIC	#177700,R2	;SETUP WRITE BUFFER VA
1588	010154	052702	020000			BIS	#20000,R2	
1589	010160	052767	140000	167610		BIS	#UMODE,PSW	;CHANGE TO USER
1590	010166	021012			18:	CMP	#R0,#R2	;DATA OK?
1591	010170	001436				BEQ	3#	;YES, BRANCH
1592	010172	005267	170732			INC	DERRCT	;NO, COUNT DATA ERROR
1593	010176	026727	170726	000003		CMP	DERRCT,#3	;MORE THAN 3 ERRORS?
1594	010204	003404				BLE	2#	;NO- REPORT ERROR
1595	010206	032777	002000	170762		BIT	#BIT10,#SR	;YES- REPORT ALL?
1596	010214	001424				BEQ	3#	;NO- BRANCH
1597	010216	011067	170672		28:	MOV	#R0,MODWPA	;TEMPORARILY SAVE DATA
1598	010222	011267	170670			MOV	#R2,MODWEA	
1599	010226	104423				RTTK#		;RETURN TO KERNEL
1600	010230	016761	170662	000056		MOV	MODWEA,ASB(R1)	;SAVE EXPECTED
1601	010236	016761	170652	000060		MOV	MODWPA,AWAS(R1)	;SAVE ACTUAL
1602	010244	10361	000054			MOV	R3,WASADR(R1)	;SAVE COUNT
1603	010250	016746	170654			MOV	DERRCT,-(SP)	;SAVE DATA ERROR COUNT
1604	010254	010346				MOV	R3,-(SP)	;SAVE WORD COUNT ON STACK
1605	010256	016746	171236			MOV	CDERR1,-(SP)	;SAVE VA OF CKDATA CALL
1606	010262	000167	171230			JMP	CDERR	;GO TO ERROR CALL
1607	010266	005203			38:	INC	R3	;COUNT WORDS CHECKED
1608	010270	005720				TST	(R0)+	
1609	010272	005722				TST	(R2)+	
1610	010274	105767	171004			TSTB	KTPRES	
1611	010300	001420				BEQ	5#	
1612	010302	032700	017777			BIT	#17777,R0	;REMAP IF NEEDED
1613	010306	001005				BNE	4#	
1614	010310	162700	020000			SUB	#20000,R0	
1615	010314	062767	000200	167322		ADD	#200,UIPAR2	
1616	010322	032702	017777		48:	BIT	#17777,R2	
1617	010326	001005				BNE	5#	
1618	010330	162702	020000			SUB	#20000,R2	
1619	010334	062767	000200	167300		ADD	#200,UIPAR1	
1620	010342	005305			58:	DEC	R5	
1621	010344	001310				BNE	1#	
1622	010346	104423				RTTK#		;RETURN TO KERNEL
1623	010350	005767	170554			TST	DERRCT	;ANY DATA ERRORS?
1624	010354	001003				BNE	6#	;YES, BRANCH
1625	010356	012467	170472			MOV	(R4)+,DSTADR	;GET THE RETURN ADDRESS
1626	010362	000423				BP	7#	;GO TO COMMON RETURN CODE
1627	010364	005261	000030		68:	INC	ERCNT(R1)	;DATA ERRORS OCCURRED= UP MODULE ERROR
1628								;COUNT BY 1
1629	010370	001002				BNE	#+6	
1630	010372	005161	000030			COM	ERCNT(R1)	;SET TO -1 IF 0
1631	010376	016761	170526	000060		MOV	DERRCT,AWAS(R1)	;STORE INFORMATION IN MODULE
1632	010404	010361	000056			MOV	R3,ASB(R1)	;STORE SIZE IN ASB
1633	010410	010446				MOV	R4,-(SP)	;SAVE POINTER TO ERROR RETURN ADDRESS
1634	010412	010661	000046			MOV	SP,SVR6(R1)	;SAVE STACK POINTER
1635	010416	104422				MSGDS		;REPORT TOTALS, HALT MODULE IF NEEDED
1636	010420	012604				MOV	(SP)+,R4	;GET ERROR RETURN POINTER

1637	010422	016701	170510			MOV	MODADR,R1	;GET MODULE START ADDRESS
1638	010426	012467	170422			MOV	(R4)+,DSTADR	;GET THE RETURN ADDRESS
1639	010432	105767	170711		78:	TSTB	DATFLG	;IS THIS A DATA CALL ?
1640	010436	100002				BPL	8#	;NO, CONTINUE
1641	010440	016734	170464			MOV	DERRCT,@(R4)+	;YES, LOAD THE NUMBER OF DATA ERRORS
1642	010444	116167	000020	170414	88:	STAT(R1),PSTAT		;SET UP FOR MODULE RETURN
1643	010452	010100				MOV	R1,R0	
1644	010454	062700	000046			ADD	#SVR6,R0	
1645	010460	000167	172422			JMP	IQGSVD	
1646								
1647	010464	005746			CDRET:	TST	-(SP)	;PUSH STACK
1648	010466	004467	004246			JSR	R4,SAV04	
1649	010472	016604	000014			MOV	14(SP),R4	;GET VA OF CKDATA CALL
1650	010476	062704	000002			ADD	#2,R4	
1651	010502	005267	170450			INC	CDFLG	
1652	010506	000167	171150			JMP	CDATA1	

```

1653 ;COMMAND DECODER, CHECKS SYNTAX AGAINST COMMAND TABLE.
1654 ;ALL COMMANDS ARE CHECKED AGAINST THE QUOTES, NO ABBREVIATIONS ALLOWED
1655
1656 010512 105767 170272 DECODE: TSTB SPCFLG ;SPECIAL FLAG SET?
1657 010516 001404 BEQ 1# ;BR IF NOT.
1658 010520 005067 170264 CLR SPCFLG ;YES, CLEAR IT.
1659 010524 000177 170326 JMP @SRETRN ;GO VIA SRETRN.
1660 010530 105767 170555 1# : TSTB RMODE ;IN RUN MODE?
1661 010534 001403 BEQ 2# ;NO, CONTINUE
1662 010536 104406 023131 MSG$,INVCD: ;'INVALID CMD. ONLY CNTL C IS OK'
1663 010542 000441 BR CONCON ;CONTINUE
1664 010544 004467 004170 2# : JSR R4,SAV04 ;SAVE REGS 0-4.
1665 010550 016700 170262 MOV KBPTR,R0 ;POINTER TO R0.
1666 010554 012701 025106 GTOK: MOV @COMTAB,R1 ;DEVICE DECODING COMES HERE
1667 010560 010046 GTOKX: MOV R0,-(SP) ;SAVE STRING POINTER
1668 010562 122021 GTOK1: CMPB (R0)+,(R1)+ ;MATCH THE CHARACTER?
1669 010564 001407 BEQ GTOK2 ;YEP, SEE IF THE REST MATCH
1670 010566 105721 TSTB (R1)+ ;NO, GO CHECK AGAINST NEXT
1671 010570 001376 BNE .-2 ;BUT FIND WHERE NEXT ENTRY IS
1672 010572 111102 MOVB (R1),R2 ;DISPLACEMENT TO NEXT ENTRY
1673 010574 001422 REQ CONCD1 ;NO SUCH THING.
1674 010576 060201 MORE: ADD R2,R1 ;POINT TO NEXT ENTRY
1675 010600 011600 MOV (SP),R0 ;POINT TO BEGINNING OF INPUT
1676 010602 000767 BR GTOK1 ;KEEP LOOKING
1677 010604 105711 GTOK2: TSTB (R1) ;LAST CHR IN ENTRY?
1678 010606 001365 RNE GTOK1 ;NO,CHECK THE OTHER CHR'S
1679 010610 122126 CMPB (R1)+,(SP)+ ;RESET SP AND R1
1680 010612 111102 MOVB (R1),R2 ;GET DISPATCH ADDRESS
1681 010614 060201 ADD R2,R1 ;R1 POINTS TO NEXT ENTRY
1682 010616 014167 170670 MOV -(R1),CADDR
1683 010622 005067 170666 CLR CSTART ;SET MODULE ADDR IN CSTART TO 0.
1684 010626 010067 170204 MOV R0,KBPTR ;SAVE CURRENT BUFFER POINTER.
1685 010632 004767 004114 JSR PC,RST04 ;RESTORE REGS 0-4.
1686 010636 000167 170640 JMP CONQUE ;GO TO COMMON QUE CALL.
    
```

```

1687 ;COMMON RETURN FROM COMMAND EXECUTION AND RUN ROUTINE (INCLUDING
1688 ;RELOCATION AND INTERNAL RESTART)
1689
1690 010642 104406 023107 COMCD1: MSG$,INVCD ;TYPE INVALID COMMAND.
1691 010646 CONCON:
1692 010646 005067 170136 COMCD2: CLR SPCFLG ;CLEAR SPECIAL FLAG AND
1693 ;DIP COMMAND INDICATOR.
1694 010652 105067 170446 CLRB FILLID ;CLEAR FILL COMMAND INDICATOR.
1695 010656 104406 022737 COMCD3: MSG$,DOT ;TYPE DOT.
1696 010662 000167 171274 COMCD4: JMP INPUT ;GO GET MORE INPUT.
    
```

```

1697 ;SPECIAL INPUT ROUTINE.
1698 ;CALLED BY MODIFY ROUTINE, WHEN WAITING FOR NEW VALUE TO BE INPUT.
1699
1700 010666 012667 170164 SINPUT: MOV (SP)+,SRETRN ;SAVE RETURN ADDR.
1701 010672 105267 170112 INCB SPCFLG ;SET SPECIAL FLAG.
1702 010676 000771 BR COMCD4
    
```

```

1703 ;RUN ROUTINE. STARTS EXERCISER EXECUTION.
1704 ;NOTE THAT "RUN" IS ENTERED ONLY FROM BANK 0, BUT "RUNRES" IS USED WHEN
1705 ;THE PROGRAM IS NOT IN BANK 0.
1706
1707 010700 112767 000001 170421 RUNL: MOVB #1,LOCK ;SET LOCK TO PREVENT RELOCATING CODE EACH
1708 ;TIME WBUF IS CYCLED
1709 010706 000402 BR .+6
1710 010710 105067 170413 RUN: CLRB LOCK ;CLEAR LOCK TO ALLOW CODE RELOCATION
1711 010714 005067 170324 CLR OFFSTD
1712 010720 005067 170340 CLR SYSCNT ;ZERO SYSTEM ERROR COUNTER
1713 010724 005067 170336 CLR PWRCNT ;ZERO POWER FAIL COUNTER
1714 010730 132767 000001 170415 6: BITB #BIT0,CLOCK ;USING THE KW11-L CLOCK ?
1715 010736 001403 BEQ 16 ;NO, CONTINUE
1716 010740 004777 170420 JSR PC,@ALCLR ;YES, MAKE SURE CLOCK TABLE IS CLEARED
1717 010744 000406 BR 29 ;CONTINUE
1718 010746 132767 000002 170377 18: BITB #BIT1,CLOCK ;USING THE KW11-P CLOCK ?
1719 010754 001402 BEQ 28 ;NO, CONTINUE
1720 010756 004777 170404 JSR PC,@APCLR ;YES, MAKE SURE CLOCK TABLE IS CLEARED
1721 010762 012767 042761 001540 28: MOV #42761,SLDSB ;RESTORE THESE INSTRUCTIONS
1722 010770 012767 042761 001554 MOV #42761,SLDSE ;FOR A CHECK SUM MODULE
1723 010776 004767 002446 JSR PC,GETADR
1724 011002 000436 BR 58
1725 011004 016701 170032 MOV ADDRLO,R1 ;IF INVALID OR NO ADDRESS, FORCE 0
1726 011010 016700 170030 MOV ADDRHI,R0 ;GET LOW ORDER BITS
1727 011014 006200 ASR R0 ;GET ANY HIGH ORDER BITS
1728 011016 006001 ROR R1
1729 011020 006200 ASR R0
1730 011022 006001 ROR R1
1731 011024 000241 CLC
1732 011026 006001 ROR R1
1733 011030 006201 ASR R1
1734 011032 006201 ASR R1
1735 011034 006201 ASP R1
1736 011036 042701 000177 BIC #177,R1 ;FORCE TO 4K BOUNDARY
1737 011042 001416 BEQ 58 ;IF NOW 0, SKIP REST (TO PREVENT ERR PRINTOUT
;IF KTPRES WAS CLEARED)
1738
1739 011044 020167 170170 CMP R1,OFFMX
1740 011050 101403 BLOS 38 ;ADDRESS IS OK, CONTINUE
1741 011052 104406 023203 MSG$,INVCD2 ;'CANNOT GO THAT HIGH'
1742 011056 000673 BR COMCON ;CONTINUE
1743 011060 105767 170220 30: TSTB KTPRES ;KT11 IN USE?
1744 011064 001003 BNE 48 ;YES, CONTINUE
1745 011066 104406 023241 MSG$,INVCD3 ;'NEED KT-11'
1746 011072 000665 BR COMCON ;CONTINUE
1747 011074 010167 170144 48: MOV R1,OFFSTD
1748 011100 112767 000001 170203 58: MOVB #1,RMODE ;MAKE SURE BANK 0 COPY OF RMODE IS
;SET WHILE EXECUTING MVMODE
1749
1750 011106 105067 170214 CLRBB RUNRFL
1751 011112 004767 010356 JSR PC,MVMODE ;MOVE PROGRAM TO NEW BANK AND GO THERE
1752 011116 105067 170167 CLRBB RMODE ;CLEAR CURRENT COPY OF RMODE
1753 011122 012767 026520 170140 MOV #MODQ-2,BKPTR
1754 011130 004767 003714 JSR PC,CLRVEC ;INITIALIZE TRAPCATCHER IN VECTOR AREA
1755 011134 004767 003626 JSR PC,CLRQUS ;INCLUDES CLEARING BRAKE,MODCNT,IOBKID
1756 011140 132767 000002 170172 BITB #BIT1,PPRES ;IS IT AN 11/60 ? *****
1757 011146 001424 BEQ 118 ;NO, CONTINUE
1758 011150 012767 100001 170316 MOV #100001,LGHOLD ;YES, LOAD ERROR LOG ENABLE BIT
    
```

```

1759                                     ;AND FREEZE ON FIRST ERROR BIT
1760 011156 WRITE RDWHMI,WTWHMI ;ENABLE 11/60 ERROR LOG
1761 011156 MOV RO,=(SP) ;SAVE REG. 0
1762 011160 076600 000022 MED,RDWHMI ;SAVE
1763 011164 056700 170304 BIS LGHOLD,RO
1764 011170 076600 000222 MED,WTWHMI
1765 011174 012600 MOV (SP)+,RO ;RESTORE REG. 0
1766 011176 017767 170274 MOV @CONTRL,KONTRL ;GET CURRENT STATE
1767 011204 052767 000200 BIS #200,KONTRL ;ADD ABORT BIT
1768 011212 016777 170262 MOV KONTRL,@CONTRL ;WRITE IT BACK *****
1769 011220 112767 177777 118: MOVB #-1,WBFLG ;INITIALIZE BUFFER ROTATION FLAG
1770 011226 012702 026522 MOVB #MODQ,R2 ;SETUP TO COUNT SELECTED MODULES, AND
                                     ;IF NOT SYSERR OR POWER FAIL RESTART
1771                                     ;CLEAR MODULES' PASCNT AND ERRCNT.
1772
1773 011232 012201 181 MOV (2)+,R1 ;MODULE ADDR TO R1.
1774 011234 001462 BEQ RUNC ;BP IF NO MORE.
1775 011236 105767 170047 TSTB RMODE ;INTERNAL RESTART?
1776 011242 001012 BNE 2# ;BR IF YES.
1777 011244 105767 170042 TSTB CHN ;IN CHAIN MODE?
1778 011250 100407 RMI 2# ;BR IF YES.
1779 011252 005061 000026 CLR PSCNT(R1) ;CLEAR MOD'S PASCNT.
1780 011256 005061 000030 CLR ERCNT(R1) ;CLEAR MOD'S ERROR COUNT.
1781 011262 042761 020000 000020 BIC #BIT13,STAT(R1) ;CLEAR STOPPED BIT.
1782 011270 032761 040000 000020 28: BIT #BIT14,STAT(R1) ;MODULE SELECTED?
1783 011276 001755 BEQ 1# ;BR IF NOT.
1784 011300 032761 020000 000020 BIT #BIT13,STAT(R1) ;MODULE STOPPED?
1785 011306 001351 BNE 1# ;BP IF YES.
1786 011310 005767 167726 TST OFFSET ;IF OFFSET 0, IGNORE BITS 10+11
1787 011314 001414 BEQ 3#
1788 011316 032761 004000 000020 BIT #BIT11,STAT(R1) ;NOT 0- IF BIT11 IS SET, DON'T COUNT IT
1789 011324 001342 BNE 1#
1790 011326 032767 001777 167706 BIT #1777,OFFSET ;32K BOUNDARY?
1791 011334 001404 BEQ 3# ;YES- IGNORE BIT 10
1792 011336 032761 002000 000020 BIT #BIT10,STAT(R1) ;NO- BIT 10 SET?
1793 011344 001332 BNE 1# ;YES- DON'T RUN THIS MODULE
1794 011346 105267 167443 38: INCB MDCNT ;NO. UP COUNT OF RUNNABLE MODULES.
1795 011352 005761 000020 TST STAT(R1) ;RKM0D?
1796 011356 100402 BMI +6 ;NO- BRANCH
1797 011360 105267 167441 INCB BKCNT ;YES- UP COUNT
1798 011364 032761 010000 000020 BIT #BIT12,STAT(R1) ;EXTENDED MODULE?
1799 011372 001402 BEQ +6 ;NO- BRANCH
1800 011374 105267 167426 INCB XCNT ;YES- COUNT IT
1801 011400 000714 BR 1# ;GO CHECK NEXT MODULE.
1802 011402 105767 167407 RUNC: TSTB MDCNT ;ANY RUNNABLE MODULES?
1803 011406 001005 BNE 5# ;YES, BRANCH
1804 011410 005767 167626 TST OFFSET ;NO, OFFSET 0?
1805 011414 001501 BEQ RUNC ;YES- COMMAND INVALID
1806 011416 000167 173572 JMP ENDSVB ;NO- THE ONLY MODS RUNNABLE MUST BE THOSE
1807                                     ;WITH BITS 10 OR 11 SET- RELOCATE
1808 011422 116767 167367 167366 58: MOVB MDCNT,MDCNTR ;LOAD COUNT OF RUNNABLE MODULES
1809                                     ;INTO STARTUP COUNTER
1810 011430 012767 026520 167402 MOV #MODQ=2,MODPTR ;MODULE TABLE ADDR TO MODPTR.
1811 011436 012767 040000 167414 MOV #40000,IOBKID ;START WITH SPECIAL BACKGROUND MODULES.
1812 011444 105267 167702 INCB NSTOP ;SET THE NON-STOP FLAG
1813 011450 105767 167637 TSTB FOTI
1814 011454 001447 BEQ 2#
    
```

```

1815 011456 116767 167344 167345 MOVB XCNT,ROTCT ;SETUP ROTATION TIMEOUT COUNTER- 16 TIMES NUMBER
1816                                     ;OF EXTENDED MODULES
1817 011464 106367 167341 ASLB ROTCT
1818 011470 106367 167335 ASLB ROTCT
1819 011474 106367 167331 ASLB ROTCT
1820 011500 106367 167325 ASLB ROTCT
1821 011504 126727 167316 000020 CMPB XCNT,#20 ;PREVENT LOSS OF COUNT IF OVERFLOW
1822 011512 103403 BLO 1#
1823 011514 112767 000377 167307 MOVB #377,ROTCT
1824 011522 116767 167303 167604 18: MOVB ROTCT,&GWBFC ;INITIALIZE WRITE BUFFER ROTATION COUNTER
1825 011530 116767 167261 167275 MOVB MDCNT,PASREL ;SET ALTERNATE RELOCATION COUNT TO 8
1826 011536 106367 167271 ASLB PASREL ;TIMES MDCNT
1827 011542 106367 167265 ASLB PASREL
1828 011546 106367 167261 ASLB PASREL
1829 011552 126727 167237 000040 CMPB MDCNT,#40
1830 011560 103403 BLO 3#
1831 011562 112767 000377 167243 MOVB #377,PASREL
1832 011570 004767 000050 38: JSR PC,SETWBF
1833 011574 105767 167511 28: TSTB RMODE
1834 011600 001005 BNE 4#
1835 011602 104406 022737 MSGS,DOT
1836 011606 112767 000001 167475 MOVB #1,RMODE ;SET RMODE IN CURRENT BANK
1837 011614 000167 170342 48: JMP INPUT
1838 011620 005067 167420 RUNC: CLR OFFSTD
1839 011624 004767 007644 JSR PC,MVCODE
1840 011630 105067 167455 CLRR RMODE
1841 011634 104406 023316 MSGS,INVCD4 ;NO MODULES SELECTED?
1842 011640 000167 177002 JMP CONCON
    
```

```

;SETUP WRITE BUFFER INFORMATION
1843
1844
1845 011644 004567 003022 SETWBF: JSR R5,TO18S
1846 011650 001242 OFFSET
1847 011652 001232 WBUF
1848 011654 001234 WBFEA
1849 011656 066767 167336 167346 ADD LOCORE,WBUF
1850 011664 103003 BCC .+10
1851 011666 062767 000020 167340 ADD #20,WBFEA
1852 011674 016767 167324 167334 MOV LCOR12,WBUF12
1853 011702 066767 167334 167326 ADD OFFSET,WBUF12
1854 011710 026767 167322 167310 CMP WBUF12,HCOR12 ;IS WRITE BUFFER AT THE TOP ?
1855 011716 001003 BNE SETWB1 ;NO, IT'S OK, CONTINUE
1856 011720 004767 004726 JSR PC,ROTLOW ;YES, SETUP TO ROTATE BELOW PROGRAM
1857 011724 000207 RTS ;RETURN
1858
1859 011726 016767 167274 167322 SETWB1: MOV HCOR12,MXWBF ;CALCULATE MAX SIZE OF WRITE BUFFER
1860 011734 105767 167352 TSTB CHN ;IN CHAIN MODE ?
1861 011740 100003 BPL 1$ ;NO, DON'T ADD ANY MORE WRITE BUFFER
1862 011742 062767 000040 167306 ADD #40,MXWBF ;YES, ADD 1K TO MAX. WRITE BUFFER SIZE
1863 011750 166767 167262 167300 1$: SUB WBUF12,MXWBF
1864 011756 006367 167274 ASL MXWBF
1865 011762 006367 167270 ASL MXWBF
1866 011766 006367 167264 ASL MXWBF
1867 011772 006367 167260 ASL MXWBF
1868 011776 006367 167254 ASL MXWBF
1869 012002 103003 BCC .+10
1870 012004 012767 177777 167244 MOV #177777,MXWBF
1871 012012 000207 RTS PC
    
```

```

;KT11 INITIALIZATION ROUTINE ASSUMES KT11 IS OFF WHEN CALLED
1872
1873
1874 012014 012737 015254 000250 KTINIT: MOV #KTERR,0#250 ;SETUP MEMORY MANAGEMENT ABORT VECTOR
1875 012022 012737 000340 000252 MOV #PRTY7,0#252 ;DON'T LET ANYTHING ABOPT A MEMORY MANAGEMENT ABORT
1876 012030 012701 001276 MOV #PDP1AB,R1 ;SETUP TO CLEAR ALL KT11 REGISTERS
1877 012034 012737 000006 000004 MOV #6,0#4 ;SETUP TO RTI IF KT11=D WHEN
1878 012042 012737 000002 000006 MOV #RTI,0#6 ;ADDRESSING REGISTERS IT LACKS
1879 012050 012703 000040 1$: MOV #32,R3 ;LOAD THE COUNT OF KT11 REGISTERS TO BE CLEARED
1880 012054 012102 MOV (R1),R2 ;GET ADDRESS OF REGISTER SET
1881 012056 005012 2$: CLR (P2) ;CLEAR ALL PAR'S AND PDR'S
1882 012060 062702 000002 ADD #2,P2 ;USE THE ADD INSTRUCTION TO POINT TO THE NEXT REGISTER.
1883 012064 005303 DEC R3 ;REDUCE COUNT OF APP REGISTERS CLEARED
1884 012066 001373 BNE 2$ ;IS COUNT EXCEEDED? IF NO, GO CLEAR AGAIN
1885 012070 020127 001302 CMP R1,#PDREND ;IF YES, THEN IS THE TABLE BOUNDARY EXCEEDED?
1886 012074 003765 BLE 1$ ;IF NO, DO THE NEXT SET IN THE TABLE
1887 012076 012737 015246 000004 MOV #0USERP,0#4 ;RESTORE THE BUS ERROR VECTOR.
1888 012104 012737 000340 000006 MOV #PRTY7,0#6 ;RESTORE THE BUS ERROR PRIORITY
1889 012112 012737 140000 177776 MOV #UMODE,0#PS ;SET TO USFR MODE
1890 012120 012706 027002 MOV #USP,SP ;LOAD USER STACK POINTER
1891 012124 105767 167203 TSTB MDL45 ;IS THIS A PDP 11/45?
1892 012130 001405 BEQ 3$ ;IF NOT, DO NOT SET UP THE SUPERVISOR STACK
1893 012132 012737 040000 177776 MOV #SMODE,0#PS ;IF YES, SET TO SUPERVISOR MODE
1894 012140 012706 027012 MOV #SSP,SP ;LOAD SUPERVISOR STACK POINTER
1895 012144 005037 177776 3$: CLR 0#PS ;RETURN TO KERNEL MODE
1896
1897 ;SET UP ALL THE PAGE DESCRIPTIONS THE SAME; READ/WRITE PROTECTION
1898 ;UPWARD PAGE ADDRESSING, AND MAXIMUM BLOCK COUNT.
1899 ; (177 OCTAL BLOCKS = 128 X 32 WORD DECTMAL BLOCKS)
1900
1901 012150 004467 002564 KTSET0: JSR R4,SAV04 ;SAVE REGS. 0 THRU 4
1902 012154 005001 CLR R1 ;MAP STARTING AT BANK 0
1903 012156 012702 172300 MOV #KIPDR0,R2 ;LOAD ADDRESS OF PDR 0
1904 012162 012703 000007 MOV #7,R3 ;LOAD COUNTER TO DO 28K
1905 012166 012712 077406 1$: MOV #77406,(R2) ;SET THE PDR TO 4K, R/W, UP
1906 012172 010162 000040 MOV R1,40(R2) ;SET THE PAR TO THE PROPER 4K BANK
1907 012176 005722 TST (R2)+ ;POINT TO NEXT PDR AND PAR
1908 012200 062701 000200 ADD #200,R1 ;STEP TO NEXT 4K BANK
1909 012204 005303 DEC R3 ;ALL DONE ?
1910 012206 001367 BNE 1$ ;NO, CONTINUE
1911
1912 012210 012737 007600 172356 MOV #7600,0#KIPAR7 ;POINT REGISTER 7 THE UNIRUS ADDRESS SPACE
1913 012216 012737 077406 172316 MOV #77406,0#KIPDR7 ;SETS UP REGISTER 7
1914 012224 004767 002522 JSR PC,RST04 ;RESTORE REGS. 0 THRU 4
1915 012230 000207 RTS PC ;RETURN TO THE CALLING PROCEDURE
    
```

```

1916                                     ;PARITY ENABLE ROUTINE --- ASSUMES THAT PARITY IS PRESENT
1917
1918 012232 132767 000002 167100 PAENBL; BITB  #BIT1,PPRES ;IS IT AN 11/60 ? *****
1919 012240 001403                                     BEQ  1# ;NO, CONTINUE
1920 012242 052777 000001 167226 B1S   #1,#CONTRL ;ENABLE CACHE PARITY TRAPS *****
1921
1922 012250 012700 001430 1#;  MOV  #PARTAB,R0 ;GET ADDRESS OF PARITY ADDRESSES
1923 012254 005001                                     CLR  R1 ;CLEAR COUNTER
1924
1925 012256 012770 000001 000000 2#;  MOV  #1,#(R0) ;SET PARITY ENABLE
1926 012264 032770 000001 000000 BIT   #1,#(R0) ;IS PARITY ENABLED ?
1927 012272 001002                                     BNE  3# ;YES, CONTINUE
1928 012274 004767 003476 JSP   PC,PAMSG ;NO, REPORT THE ERROR
1929
1930 012300 005720 3#;  TST  (R0)+ ;STEP TO NEXT REGISTER ADDRESS
1931 012302 005201 TNC  R1 ;STEP THE REGISTER COUNTER
1932 012304 126701 1#^031 CMPB  CNT1,R1 ;ALL PARITY REGISTERS SET ?
1933 012310 003362 BGT  2# ;NO, DO THE NEXT ONE
1934
1935 012312 000207 4#;  RTS  PC ;YES, ALL DONE, RETURN
1936
    
```

```

1937                                     ;MAP ROUTINE, TYPES RESIDENT MODULES AND THEIR START ADDRESS,
1938
1939 012314 105267 166471 MAP;  INCB  DIRIND ;SET DIR INDICATOR,
1940 012320 105067 013303 CLRB  MDIRE ;TERMINATE ASCII STRING EARLY,
1941 012324 004767 000004 JSR   PC,DIRA ;TYPE MAP,
1942 012330 000167 176312 JMP   COMCON
1943
1944 012334 012702 026522 DIRA;  MOV  #MODQ,R2 ;GET MODULE TABLE ADDR,
1945 012340 012201 1#;  MOV  (2)+,R1 ;GET MODULE ADDR,
1946 012342 001446 BEQ   5# ;RR IF 0, ALL DONE,
1947 012344 032761 040000 000020 BIT   #BIT14,STAT(R1) ;MODULE SELECTED?
1948 012352 001003 BNE  2# ;RR IF YES,
1949 012354 105767 166431 TSTB  DIRIND ;TYPING DIRECTORY?
1950 012360 004767 BEQ   1# ;RR IF NOT, DONT TYPE UNSSELECTED MODS,
1951 012362 004567 001246 JSR   R5,FILLNM ;FILL MOD NAME IN ASCII STRING,
1952 012366 025574 AMODNM+1 ;ADDR TO STUFF NAME IN,
1953 012370 010146 MOV   R1,-(SP) ;MODULE ADDR TO STACK,
1954 012372 004567 001542 JSR   R5,OACNV ;CONVERT MOD ADDR TO ASCII,
1955 012376 025605 APC
1956 012400 016146 000020 MOV   STAT(R1),-(SP) ;CONVERT MODULE STATUS,
1957 012404 004567 001530 JSR   R5,OACNV
1958 012410 025621 AMDSTA
1959 012412 016146 000026 MOV   PSCNT(R1),-(SP) ;MOD'S PASS COUNT TO STACK,
1960 012416 004567 001750 JSR   R5,BDCNV ;CONVERT IT TO DECIMAL ASCII,
1961 012422 025640 APSCNT
1962 012424 016146 000030 MOV   ERCNT(R1),-(SP) ;MOD'S ERROR COUNT TO STACK,
1963 012430 004567 001736 JSR   R5,RDCNV ;CONVERT IT TO DECIMAL ASCII,
1964 012434 025557 AERRS
1965 012436 105767 166347 TSTR  DIRIND ;TYPING DIRECTORY?
1966 012442 001003 BNE  3# ;RR IF YES,
1967 012444 112767 000040 013155 MOVB  #40,MDIRE ;NO, ALLOW TYPING FULL STRING,
1968 012452 104406 025573 3#;  MSGS,AMODNM ;TYPE ASCII LINE,
1969 012456 000730 BR    1# ;DO IT AGAIN,
1970 012460 000207 5#;  RTS  PC ;EXIT,
    
```

```

1971                                     ;SELECT MODULE(S) ROUTINE,
1972
1973 012462 012767 052761 000040 SEL:  MOV  #52761,SLDSB ;MODIFY COMMON TO SELECT MODULE(S),
1974 012470 012767 052761 000054     MOV  #52761,SLDSE ;
1975 012476 000406                     BR    SLDS ;GO TO COMMON,
1976
1977                                     ;DESELECT MODULE(S) ROUTINE,
1978
1979 012500 012767 042761 000022 DES:  MOV  #42761,SLDSB ;MODIFY COMMON TO DESELECT MODULE(S),
1980 012506 012767 042761 000036     MOV  #42761,SLDSE
1981
1982                                     ;SELECT/DESELECT COMMON,
1983
1984 012514                               SLDS:
1985 012514 004767 001142             SLDSA: JSR  PC,GETNAM ;CHECK NAME, GET MODULE ADDR,
1986 012520 000410                     BR    SLDSL ;NO NAME, SELECT/DESELECT ALL,
1987 012522 000405                     BR    SLDSL ;INVALID OR NEX NAME,
1988 012524 016701 166310             MOV  MDPTR,R1 ;MODULE ADDR TO R1,
1989 012530 042761 040000 000020 SLDSE: BIC  #BIT14,STAT(R1) ;SELECT/DESELECT MODULE,
1990 012536 000167 176104             SLDSE: JMP  COMCON ;DONE,
1991 012542 012702 026522             SLDSE: MOV  #MODQ,R2 ;MODULE TABLE ADDR TO R2,
1992 012546 012201                     SLDSE: MOV  (2),R1 ;GET MODULE ADDR,
1993 012550 001772                     SLDSE: BEQ  SLDSL ;BR IF 0,END OF TABLE, DONE,
1994 012552 042761 040000 000020 SLDSE: BIC  #BIT14,STAT(R1) ;SELECT/DESELECT MODULE,
1995 012560 000772                     BR    SLDSL ;DO IT AGAIN,
1996                                     ;LOCATIONS SLDSB AND SLDSE ARE PURE WHILE IN RUN MODE,
    
```

```

1997                                     ;KT11 DISABLE ROUTINE
1998
1999 012562 105767 166516             KTOFF: TSTB KTPRES
2000 012566 001002                     BNE  18 ;
2001 012570 000167 176062             18:  JMP  COMC03 ;IGNORE COMMAND- NO KT OP ALREADY OFF
2002 012574 105067 166504             18:  CLRB KTPRES
2003
2004
2005 012600 005037 177572             CLR  #SSRO ;TURN OFF KT11
2006 012604 026727 166416 001600     CMP  HCOR12,#1600 ;MORE THAN 28K AVAILABLE ?
2007 012612 101403                     RLOS 28 ;NO, CONTINUE
2008 012614 012767 001600 166404     MOV  #1600,HCOR12 ;YES, SET MEMORY SIZE TO 28K
2009 012622 104406 024007 28:      MSGS,KTDIS ;"KT11 OFF"
2010 012626 000167 176024             JMP  COMC03
2011
2012                                     ;KT11 ENABLE ROUTINE
2013
2014 012632 012737 012704 000004 KTON:  MOV  #13,#4
2015 012640 032737 000001 177572     BIT  #1,#SSRO ;ALREADY ON ?
2016 012646 001027                     BNE  28 ;YES, FORGET IT
2017 012650 112767 000001 166426     MOVR #1,KTPRES ;NO, SET KT PRESENT FLAG
2018 012656 016767 166346 166342     MOV  HCORSV,HCOR12 ;RESTORE MEMORY SIZE
2019 012664 004767 177124             JSR  PC,KTINIT ;INITIALIZE KT11
2020 012670 012737 000001 177572     MOV  #1,#SSRO ;TURN IT ON
2021
2022
2023 012676 104406 024615             MSGS,KTONM ;"KT11 ENABLED"
2024 012702 000411                     BR    28 ;CONTINUE
2025 012704 012737 015246 000004 18:  MOV  #BUSERR,#4
2026 012712 005067 165060             CLR  PSW
2027 012716 104406 023776             MSGS,NOKT ;"NO KT11"
2028 012722 000167 175720             JMP  COMCON ;CONTINUE
2029 012726 012737 015246 000004 28:  MOV  #BUSERR,#4
2030 012734 005067 165036             CLR  PSW
2031 012740 000167 175712             JMP  COMC03
    
```

```

2032
2033
2034 012744 105067 166344
2035 012750 004767 003764
2036 012754 000167 175676
2037
2038
2039
2040
2041 012760 112767 000377 166326
2042 012766 016767 166326 166322
2043 012774 104406 024555
2044 013000 000167 175652

;THIS ROUTINE ENABLES WRITE BUFFER ROTATION
ROTON: CLRB ROTL ;CLEAR ROTATION LOCK FLAG
        JSR PC,ROCHK ;GO DO "ENABLED" MESSAGE
        JMP COMCO3 ;CONTINUE

;THIS ROUTINE DISABLES WRITE BUFFER ROTATION
ROTOFF: MOVB #377,ROTL ;SET THE ROTATION LOCK FLAG
        MOV ROTNUM,ROTCNT ;LOAD THE ROTATION COUNTER
        MSG#,ROTDIS ;"WBUF ROTATION DISABLED"
        JMP COMCO3 ;CONTINUE
    
```

```

2045
2046
2047 013004 004767 000004
2048 013010 000167 175642
2049
2050
2051
2052
2053
2054 013014 132767 000001 166316
2055 013022 001405
2056 013024 004767 177202
2057 013030 104406 023756
2058 013034 000402
2059 013036 104406 024021
2060 013042 000207
2061
2062
2063
2064
2065 013044 132767 000001 166266
2066 013052 001003
2067 013054 104406 024021
2068 013060 000421
2069 013062 132767 000002 166250
2070 013070 001403
2071 013072 042777 000001 166376
2072
2073 013100 012700 001430
2074 013104 005001
2075
2076 013106 005030
2077 013110 005201
2078 013112 126701 166223
2079 013116 003373
2080
2081 013120 104406 024034
2082 013124 000167 175526
2083
2084
2085
2086 013130 010046
2087 013132 012700 000200
2088 013136 076600
2089 013140 000352
2090 013142 012600
2091 013144 042767 000014 166326
2092 013152 016777 166322 166316
2093 013160 104406 023643
2094 013164 000167 175466
2095
2096
2097
2098
2099 013170 052767 000014 166302
2100 013176 016777 166276 166272

;THIS ROUTINE IS NECESSARY TO SUPPORT THE PON COMMAND
PON: JSR PC,PONN ;GO TURN ON PARITY
     JMP COMCO3 ;CONTINUE

;THIS ROUTINE WILL TURN ON PARITY IF AVAILABLE
PONN: BITB #BIT0,PPRES ;ANY PARITY TO TURN ON ?
      BEQ 1$ ;NO, SAY SO AND RETURN
      JSR PC,PAENBL ;YES, GO TURN IT ON
      MSG#,PENABL ;"PARITY ENABLED"
      RR 2$ ;RETURN
1$: MSG#,NOPAR ;"NO PARITY"
2$: RTS PC ;RETURN

;THIS ROUTINE DISABLES PARITY IF PRESENT
POFF: BITB #BIT0,PPRES ;ANY PARITY TO TURN OFF ?
      BNE 1$ ;YES, CONTINUE
      MSG#,NOPAR ;"NO PARITY"
      RR 4$ ;RETURN
1$: BITB #BIT1,PPRES ;IS IT AN 11/60 ? *****
      BEQ 2$ ;NO, CONTINUE
      BIC #1,#CONTRL ;DISABLE CACHE PARITY TRAPS *****
2$: MOV #PARTAB,R0 ;GET ADDRESS OF PARITY ADDRESSES
     CLR R1 ;CLEAR THE COUNTER
3$: CLR @(R0)+ ;CLEAR THE PARITY REGISTER
     INC R1 ;STEP THE REGISTER COUNTER
     CMPB CNT1,R1 ;ALL PARITY REGISTERS CLEARED ?
     BGT 3$ ;NO, CLEAR THE NEXT ONE
4$: MSG#,PAOFF ;"PARITY OFF"
     JMP COMCO3 ;CONTINUE

;THIS ROUTINE TURNS ON THE CACHE. SUPPORTS THE CON COMMAND.
CON: MOV R0,=(SP) ;SAVE R0
     MOV #200,R0 ;SWEEP CACHE CLEAN FIRST
     MED 352
     MOV (SP)+,R0 ;RESTORE R0
     BIC #14,KONTRL ;CLEAR OUT THE FORCE MISS BITS
     MOV KONTRL,@CONTRL ;DO IT
     MSG#,CONN ;"CACHE ENABLED"
     JMP COMCO3 ;CONTINUE

;THIS ROUTINE TURNS OFF THE CACHE. SUPPORTS THE COFF COMMAND.
COFF: BIS #14,KONTRL ;SET THE FORCE MISS BITS
      MOV KONTRL,@CONTRL ;DO IT
    
```

2101 013204 104406 023662  
 2102 013210 000167 175442  
 2103

MSGs,COFFF  
 JMP COMCO3

;"CACHE DISABLED"

```

2104                                     ;MODIFY ROUTINE.
2105                                     ;GETS ADDRESS AND PRINTS ADDR CONTENTS.
2106                                     ;USER CAN INPUT NEW NUMBER OR CR.
2107                                     ;CR MEANS HE JUST WANTS TO EXAMINE.
2108
2109 013214 112767 177777 166102  FILL:  MOVR  #-1,FILLID  ;SET FILL INDICATOR.
2110 013222  MOD:
2111 013222 004767 000434  58:  JSR   PC,GETNAM  ;GET NAME AND MODHUF ADDR.
2112 013226 000401  BR    48          ;NO NAME.
2113 013230 000440  BR    38          ;INVALID OR NEX NAME.
2114 013232 004767 000212  48:  JSR   PC,GETADR  ;GET ADDRESS.
2115 013236 000430  BR    66          ;INVALID ADDR OR NO ADDR.
2116 013240 016704 165576  MOV   ADDRLO,R4  ;MOVE ADDR TO R4.
2117 013244 066704 165570  ADD   MODPTR,R4  ;ADD MODULE ADDR TO ADDR.
2118 013250 010446  18:  MOV   R4,-(SP)  ;TYPE ADDR.
2119 013252 004767 000150  JSR   PC,ITOA    ;
2120 013256 011446  78:  MOV   (R4),-(SP) ;GET CONTENTS OF ADDR.
2121 013260 004767 000142  JSR   PC,ITOA    ;PRINT IT OUT.
2122 013264 004767 175376  JSR   PC,SINPUT  ;GET NEW CONTENTS.
2123 013270 004767 000222  JSR   PC,GETNUM  ;CONVERT INPUT TO BINARY.
2124 013274 000416  BR    36          ;INVALID DATA (NOT A NUMBER).
2125 013276 005767 165560  TST   YES       ;NUMBER?
2126 013302 001401  BEQ   28         ;BR IF NOT.
2127 013304 010114  MOV   R1,(R4)   ;STORE NEW VALUE.
2128 013306 122702 000012  28:  CMPB  #12,R2    ;LINE FEED?
2129 013312 001007  BNE   36        ;NO, DONE.
2130 013314 005724  TST   (R4)+     ;POINT TO NEXT LOCATION.
2131 013316 000754  BR    16        ;REPEAT FOR NEXT LOC.
2132 013320 012704 001306  68:  MOV   #FILCNT,R4 ;GET FILCNT ADDR.
2133 013324 105767 165774  TSTB  FILLID    ;IN FILL MODE?
2134 013330 001352  BNE   78        ;BR IF YES.
2135 013332 000167 175310  38:  JMP   COMCON    ;DONE.
    
```

```

2136                                     ;BINARY TO ASCII TYPE ROUTINE, TYPES NUMBER(S) ON STACK,
2137
2138 013336 016646 000002                ITOAX: MOV 2(SP),=(SP)      ;LOAD EA BITS
2139 013342 016666 000002 000004        MOV 2(SP),4(SP)
2140 013350 016666 000006 000002        MOV 6(SP),2(SP)
2141 013356 004567 000570                JSR R5,OACNVX
2142 013362 025675                        HOLD6                               ;AND PUT IT HERE
2143 013364 104406 025675                MSG$,HOLD6                           ;TYPE OUT THE VALUE
2144 013370 000425                        BR ITOUT
2145
2146 013372 016646 000002                ITOA8: MOV 2(SP),=(SP)      ;LOAD UPPER BITS
2147 013376 016666 000002 000004        MOV 2(SP),4(SP)                      ;AND LOWER BITS
2148 013404 016666 000006 000002        MOV 6(SP),2(SP)                      ;AND ADJUST THE STACK
  
```

```

2149 013412 004567 000542                JSR R5,OACNV8                          ;CONVERT 8 OCTAL NUMBERS TO ASCII
2150 013416 025706                        HOLD8                               ;AND PUT THEM HERE
2151 013420 104406 025706                MSG$,HOLD8                           ;TYPE OUT THE OCTAL VALUE
2152 013424 000407                        BR ITOUT                              ;GO TYPE IT OUT
2153
2154
2155 013426 016646 000002                ITOA1: MOV 2(SP),=(SP)      ;CONVERT NUMBER TO ASCII,
2156 013432 004567 000502                JSR R5,OACNV                          ;CONVERT NUMBER TO ASCII,
2157 013436 025675                        HOLD6                               ;TYPE OCTAL VALUE.
2158 013440 104406 025675                MSG$,HOLD6
2159 013444 012616                        ITOUT: MOV (SP)+,(SP)
2160 013446 000207                        RTS PC                                ;EXIT,
  
```

```

2161                                     ;THIS ROUTINE GETS THE ADDRESS ASSOCIATED WITH THE MOD COMMAND
2162
2163 013450 004767 000042          GETADR: JSR      PC,GETNUM          ;GET NUMBER
2164 013454 000414                                     BR      16          ;INVALID DATA RETURN,
2165 013456 005767 165400          TST      YES          ;NUMBER?
2166 013462 001411          BEQ      16          ;BR IF NOT,
2167 013464 006201          ASP      R1          ;GOOD ADDRESS?
2168 013466 103410          BCS      ADRERR          ;NOGOOD, IT'S OPD
2169 013470 060101          ADD      R1,R1          ;RESTORE THE ADR
2170 013472 010167 165344          MOV      R1,ADDRLO          ;STORE AT ADDRLO
2171 013476 010367 165342          MOV      R3,ADDRHI          ;AND ANY HIGH ORDER BITS IN ADDRHI
2172 013502 062716 000002          ADD      #2,(SP)          ;SET UP GOOD EXIT,
2173 013506 000207          18:      RTS      PC          ;DONE
2174
2175 013510 104406 023344          ADRERR: MSG6,INVADR          ;TYPE INVALID ADDR OR DATA,
2176 013514 000207          RTS      PC          ;EXIT,
    
```

```

2177                                     ;GET NUMBER FROM KEYBOARD BUFFER, AND RETURN THE LOW ORDER BITS
2178                                     ;IN R1 AND THE HIGH ORDER BITS IN R3.
2179                                     ;ROUTINE ADDS 2 TO THE RETURN PC IF SUCCESSFUL.
2180
2181 013516 016700 165314          GETNUM: MOV      RBPTR,R0          ;GET BUFFER POINTER,
2182 013522 005001          38:      CLR      R1          ;DATA
2183 013524 005003          CLR      R3
2184 013526 005067 165330          CLR      YES          ;FLAG
2185 013532 112002          18:      MOVB    (P0)+,R2          ;GET A BYTE
2186 013534 120227 000040          CMPB    R2,#40          ;SPACE?
2187 013540 001770          BEQ      38          ;YES,IGNOPE IT
2188 013542 122702 000015          CMPB    #15,R2          ;CR?
2189 013546 001425          BEQ      26          ;YES,RETURN
2190 013550 120227 000012          CMPB    R2,#12          ;LF?
2191 013554 001422          BEQ      26          ;DONE
2192 013556 120227 000060          CMPB    R2,#0          ;LOW LIMIT
2193 013562 002752          BLT      ADRERR          ;TOO LOW, REPORT THE ERROR
2194 013564 120227 000067          CMPB    R2,#7          ;HIGH LIMIT
2195 013570 003347          BGT      ADRERR          ;TOO HIGH, REPORT THE ERROR
2196 013572 006301          ASL      R1          ;SHIFT OLD STUFF
2197 013574 006103          ROL      R3
2198 013576 006301          ASL      R1          ;3 TIMES LEFT
2199 013600 006103          ROL      R3
2200 013602 006301          ASL      R1
2201 013604 006103          ROL      R3          ;I.E. MULT. BY OCTAL 10
2202 013606 060201          ADD      R2,R1
2203 013610 162701 000060          SUB      #0,R1          ;ADD NEW TO OLD
2204 013614 005267 165242          INC      YES          ;BUT GET RID OF ASCII STUFF
2205 013620 000744          BP      16          ;SET FLAG
2206 013622 010067 165210          26:      MOV      R0,RBPTR          ;MORE, MORE
2207 013626 062716 000002          ADD      #2,(SP)          ;SAVE CURRENT BUF POINTER,
2208 013632 000207          RTS      PC          ;SET UP OK EXIT,
                                     ;EXIT,
    
```

```

2209 ;THIS ROUTINE PUTS THE MODULE NAME INTO AN ASCII STRING FOR OUTPUT
2210
2211 013634 010346 FILLNM: MOV R3,-(SP) ;SAVE R3.
2212 013636 012503 MOV (R5)+,R3 ;STORE ADDR TO R3.
2213 013640 012704 000005 MOV #5,R4 ;WILL DO 5 TIMES.
2214 013644 112123 18: MOV (R1)+,(R3)+ ;MOVE CHAR.
2215 013646 005304 DFC R4 ;DONE?
2216 013650 001375 BNE 18 ;BR IF NOT.
2217 013652 162701 000005 SUB #5,R1 ;RESTORE R1.
2218 013656 012603 MOV (SP)+,R3 ;RESTORE R3.
2219 013660 000205 RTS R5 ;EXIT.
    
```

```

2220 ;THIS ROUTINE GETS THE MODULE NAME ASSOCIATED WITH THE MOD AND SEL/DES COMMANDS.
2221
2222 013662 016700 165150 GETNAM: MOV KBPTR,R0 ;GET BUFFER POINTER.
2223 013666 005067 165170 CLR YES ;CLEAR SUCCESS INDICATOR.
2224 013672 005067 165142 CLR MODPTR ;IF 0, ABS ADDR.
2225 013676 012702 000005 MOV #5,R2 ;WILL DO 5 TIMES.
2226 013702 012701 025574 MOV #AMODNM+1,R1 ;WILL STORE KYBD NAME IN TNPNAM.
2227 013706 112721 000040 18: MOV #40,(R1)+ ;BUT CLEAR IT TO SPACES FIRST.
2228 013712 005302 DEC R2 ;DONE?
2229 013714 001374 BNE 18 ;BR IF NOT.
2230 013716 012701 025574 MOV #AMODNM+1,R1 ;RELOAD TEMP ADDR.
2231 013722 111002 28: MOV (R0),R2 ;GET CHAR FROM KBUF AREA.
2232 013724 122702 000015 CMPE #15,R2 ;CR?
2233 013730 004442 BEQ 68 ;BR IF YES.
2234 013732 122702 000012 CMPE #12,R2 ;LF? TREAT AS CR.
2235 013736 001437 BEQ 68 ;BR IF YES.
2236 013740 122702 000040 CMPE #40,R2 ;SPACE? DISREGARD LEADING ONES.
2237 013744 001430 BEQ 58 ;BR IF YES.
2238 013746 120227 000101 CMPE R2,#'A ;A OR HIGHER?
2239 013752 002403 BLT 38 ;BR IF LESS. (COULD BE A NUMBER).
2240 013754 120227 000132 CMPE R2,#'Z ;Z OR LESS?
2241 013760 003411 BLE 48 ;BR IF Z OR LESS.
2242 013762 120227 000060 38: CMPE R2,#'0 ;0 OR HIGHER?
2243 013766 002427 BLT 78 ;BR IF LESS, YOU LOOSE!
2244 013770 120227 000071 CMPE R2,#'9 ;9 OR LESS?
2245 013774 003024 BGT 78 ;BR IF GREATER, YOU LOOSE!.
2246 013776 005767 165060 TST YES ;ANY PREVIOUS CHARS?
2247 014002 001426 BEQ 88 ;BR IF NOT, NOT LEADING NUMERICS ALLOWED!.
2248 014004 026727 165052 48: CMP YES,#5 ;5TH CHARACTER?
2249 014012 002015 BGE 78 ;IF YES, BRANCH
2250 014014 110221 MOV (R1)+,R2 ;STORE CHAR.
2251 014016 005267 165040 INC YES
2252 014022 005200 418: INC R0 ;POINT TO NEXT CHAR.
2253 014024 000736 BR 28 ;DO IT AGAIN!
2254 014026 005767 165030 58: TST YES ;ANY CHARS?
2255
2256
2257
2258 014032 001773 BEQ 418 ;BR IF NOT.
2259 014034 000416 BR 98
2260
2261 014036 005767 165020 68: TST YES ;ANY CHARS?
2262 014042 001013 BNE 98 ;BR IF YES.
2263 014044 000405 BR 88 ;NO NAME.
2264 014046 104406 023370 78: MSG#,INVNAM ;TYPE INVALID OR NEX (NON-EXISTENT) MESSAGE.
2265 014052 062716 000002 ADD #2,(SP) ;SET UP INVALID NAME EXIT.
2266 014056 000207 RTS PC ;EXIT.
2267
2268
2269 014060 005067 164776 88: CLR YES ;
2270 014064 010067 164746 MOV R0,KBPTR ;SAVE CURRENT BUF POINTER.
2271 014070 000207 RTS PC ;EXIT.
2272 014072 012702 026522 98: MOV #MODG,R2 ;MODULE TABLE ADDR TO R2.
2273 014076 012203 108: MOV (R2)+,R3 ;GET MODULE ADDR.
2274 014100 001762 BEQ 78 ;INVALID NAME, NO MODULE!
2275 014102 012701 025574 MOV #AMODNM+1,R1
    
```

2276	014106	012704	000005		MOV	#5,R4	;WILL MATCH 5 CHARS.
2277	014112	122123		118:	CMPB	(R1)+,(R3)+	;CHAR MATCH?
2278	014114	001370			BNE	108	;BR IF NOT.
2279	014116	005304			DEC	R4	;DONE?
2280	014120	001374			BNE	118	;BR IF NOT.
2281	014122	014267	164712		MOV	=(R2),MODPTR	;MODULE ADDR TO MODPTR.
2282	014126	010067	164704		MOV	R0,KBPTR	;SAVE CURRENT BUF POINTER.
2283	014132	062716	000004		ADD	#4,(SP)	;SET UP SUCCESS EXIT.
2284	014136	000207			RTS	PC	;EXIT.

2285							;OCTAL TO ASCII CONVERT ROUTINE.
2286							;CONVERTS ARGUMENT ON STACK TO ASCII, STORES STRING STARTING AT ADDRESS
2287							;GIVEN FOLLOWING THE CALL
2288							;EXTENDED ENTRY REQUIRES ADDITIONAL ARGUMENT ON STACK- AFTER
2289							;PUSHING LOWER 16 BITS ONTO STACK, PUSH EA BITS (SHIFTED TO POS 4,5)
2290							
2291	014140	005067	164746		DACNV:	CLR OASCEA	;CLEAR OUT UPPER BITS -- NOT USED
2292	014144	105067	165172		CLRB	ATE	;CLEAR 8 CHARACTER FLAG
2293	014150	000412			BR	CONVRT	;CONTINUE
2294							
2295	014152	105067	165164		DACNVX:	CLRB ATE	;CLEAR 8 CHARACTER FLAG
2296	014156	000403			BR	HOP	;CONTINUE
2297							
2298	014160	112767	000200	165154	DACNV8:	MOV8 #BIT7,ATE	;SET 8 CHARACTER FLAG
2299							
2300	014166	016667	000002	164716	HOP:	MOV 2(SP),OASCEA	
2301	014174	012616			MOV	(SP)+,(SP)	
2302	014176	004467	000536		CONVRT:	JSR R4,SAV04	;SAVE REGS 0-4.
2303	014202	016600	000014		MOV	12,(SP),R0	;GET OCTAL VALUE.
2304	014206	012501			MOV	(R5)+,R1	;GET DEST ADDR.
2305	014210	012702	000006		MOV	#6,R2	;SET CONVERT COUNT TO 6.
2306	014214	060201			ADD	R2,R1	;DEVELOP ADDR TO STORE 1ST CHAR.
2307	014216	105767	165120		TSTB	ATE	;SET UP TO DO 8 CHARACTERS ?
2308	014222	100002			BPL	38	;NO, CONTINUE
2309	014224	062701	000002		ADD	#2,R1	;YES, ADD 2 TO THE ADDRESS
2310	014230	010003		38:	MOV	R0,R3	;GET VALUE TO P3.
2311	014232	042703	177770		BIC	#177770,R3	;ISOLATE LEAST SIGNIFICANT DIGIT.
2312	014236	062703	000060		ADD	#6,R3	;CONVERT TO ASCII.
2313	014242	110341			MOV8	R3,=(R1)	;STORE IT.
2314	014244	042700			BIC	#7,R0	;CLEAR DIGIT JUST CONVERTED.
2315	014250	006000	000007		ROR	R0	;SHIFT IN NEXT DIGIT.
2316	014252	006000			ROR	R0	
2317	014254	006000			ROR	R0	
2318	014256	005302			DEC	R2	;DONE 6 DIGITS?
2319	014260	001363			BNE	38	;BR IF NOT.
2320	014262	105767	165054		TSTB	ATE	;SET UP TO DO 8 CHARACTERS ?
2321	014266	100410			BMI	48	;YES, DO IT
2322	014270	016702	164616		MOV	OASCEA,R2	
2323	014274	006202			ASR	R2	
2324	014276	006202			ASR	R2	
2325	014300	006202			ASR	R2	
2326	014302	060203			ADD	R2,R3	
2327	014304	110311			MOV8	R3,(R1)	
2328	014306	000425			BR	58	;GO FINISH UP
2329							
2330	014310	132767	000001	165024	48:	BITB #BIT0,ATE	;ALL DONE ?
2331	014316	001021			BNE	58	;YES, GET OUT
2332	014320	016700	164566		MOV	OASCEA,R0	;NO, LOAD UPPER NUMBERS TO BE CONVERTED
2333	014324	006300			ASL	R0	;LINE UP BITS BELONGING TO LOWER NUMBER
2334	014326	042700	177770		BIC	#177770,R0	;GET RID OF ALL OTHERS
2335	014332	060003			ADD	R0,R3	;ADD IT IN TO COMPLETE THE NUMBER
2336	014334	110311			MOV8	R3,(R1)	;STORE IT
2337	014336	016700	164550		MOV	OASCEA,R0	;LOAD UPPER BITS AGAIN
2338	014342	006200			ASR	R0	;GET RID OF THE 2 BITS
2339	014344	006200			ASR	R0	;JUST TAKEN CARE OF
2340	014346	012702	000002		MOV	#2,R2	;SET UP TO DO THE LAST 2 CHARACTERS

```

2341 014352 152767 000001 164762      BISH #BITO,ATE ;SET 'ALL DONE' FLAG
2342 014360 000723          BR      36 ;GO DO THE LAST 2 CHARACTERS
2343
2344 014362 004767 000364      56: JSR   PC,RST04 ;RESTORE REGS. 0 THRU 4
2345 014366 012616          MOV   (SP)+,(SP) ;
2346 014370 000205          RTS   R5 ;RETURN
    
```

```

2347 ;BINARY TO DECIMAL ASCII CONVERT ROUTINE.
2348 ;CONVERTS ARGUMENT ON STACK TO DECIMAL ASCII, STORES IT
2349 ;STARTING AT ADDRESS GIVEN AFTER THE CALL.
2350 ;REMOVES ARGUMENT FROM STACK BEFORE RETURNING
2351
2352 014372 004467 000342      RDCNV: JSR   R4,SAV04 ;SAVE REGS 0-4.
2353 014376 016601 000014      MOV   12,(SP),R1 ;GET BIN VALUE.
2354 014402 012700 026513      MOV   #DECVAL,R0 ;GET ADDR OF DECVAL STRING.
2355 014406 012702 014472      MOV   #TENPWR,R2 ;ADDR OF TENPWR TO R2.
2356 014412 012703 000005      MOV   #5,R3 ;SET UP TO DO 5 CONVERSIONS.
2357 014416 005004      18: CLR   R4 ;CLEAR RESULT.
2358 014420 161201      28: SUB   (R2),R1 ;SUBTRACT TEN POWER.
2359 014422 103402      BCS   38 ;BR IF UNSUCCESSFUL.
2360 014424 005204      INC   R4 ;ADD 1 TO RESULT.
2361 014426 000774      BF    28 ;DO IT AGAIN.
2362 014430 062201      38: ADD   (R2)+,R1 ;RESTORE SUBTRACTED VALUE.
2363 014432 062704 000060      ADD   #60,R4 ;MAKE IT ASCII.
2364 014436 110420      MOVR  R4,(R0)+ ;SAVE IT.
2365 014440 005303      DEC   R3 ;DONE 5?
2366 014442 001365      BNE   18 ;BR IF NOT.
2367 014444 012501      MOV   (R5)+,R1 ;GET FINAL STORE ADDR.
2368 014446 012702 000005      MOV   #5,R2 ;GET DIGIT COUNT DESIRED.
2369 014452 060201      ADD   R2,R1 ;COMPUTE ADDR OF 1ST DIGIT.
2370 014454 114041      48: MOVB -(R0),-(R1) ;TRANSFER CHAR.
2371 014456 005302      DEC   R2 ;DONE?
2372 014460 001375      BNE   48 ;BR IF NOT.
2373 014462 004767 000264      JSR   PC,RST04 ;RESTORE REGS 0-4.
2374 014466 012616          MOV   (SP)+,(SP) ;
2375 014470 000205          RTS   R5 ;EXIT.
2376 014472 023420 001750 000144 TENPWR: 10000.,1000.,100.,10.,1
2377 014500 000012 000001
    
```

```

2378 ;RETURN TO KERNEL MODE
2379
2380 014504 042766 170000 000002 RTTK.1 BIC #170000,2(SP)
2381 014512 000002 RTI
2382
2383 ;
2384 ; ROUTINE TO CALC NEXT RANDOM NUMBER, USES A 29 BIT SHIFT REGISTER
2385 ; ALGORITHM, EXCLUSIVELY OR-ING THE 28TH AND 1ST(LOW ORDER) BITS
2386 ; INTO THE 29TH BIT AS IT SHIFTS RIGHT, SHIFTS 16 TIMES EACH TIME
2387 ; ITS CALLED. MODULES DO A RAND CALL THEN JUST REFERENCE RANNUM
2388 ; WHICH IS DEFINED AS GLOBAL IN DDXCOM
2389 ;
2390 014514 004467 000220 RAND.1 JSR R4,SAV04 ;SAVE THE REGISTERS
2391 014520 016700 164456 MOV RANWRK,R0 ;GET UPPER BITS
2392 014524 016701 164450 MOV RANNUM,R1 ;GET LOWER ORDER 16 BITS
2393 014530 012702 000020 MOV #16.,R2 ;SHIFT 16 TIMES
2394 014534 006000 RLOOP: ROR R0 ;SHIFT R0 INTO CARRY
2395 014536 006001 ROR R1 ;SHIFT CARRY INTO R1, R1 INTO CARRY
2396 014540 103410 BCS 18 ;FIRST HALF OF EX-OR PROCESS
2397
2398 014542 042700 010000 BIC #10000,R0 ;ASSUME OLD BIT 28 WAS ALSO CLEAR
2399 014546 032700 002000 BIT #2000,R0 ;CHECK THAT ASSUMPTION
2400 014552 001412 BEQ 28 ;BP IF RIGHT, ELSE
2401 014554 052700 010000 BIS #10000,R0 ;PUT BIT IN
2402 014560 000407 BR 28 ;GO SHIFT AGAIN
2403
2404 014562 052700 010000 18: BIS #10000,R0 ;ASSUME OLD BIT 28 WASN'T ALSO SET
2405 014566 032700 002000 BIT #2000,R0 ;CHECK THAT ASSUMPTION
2406 014572 001402 BEQ 28 ;BR IF RIGHT
2407 014574 042700 010000 BIC #10000,R0 ;ELSE 2 1'S = A ZERO BIT
2408
2409 014600 005302 28: DEC R2 ;DONE 16 TIMES?
2410 014602 001354 BNE RLOOP ;BR IF NO
2411 014604 010067 164372 MOV R0,RANWRK ;SAVE CURRENT VALUES
2412 014610 010167 164364 MOV R1,RANNUM
2413 014614 004767 000132 JSR PC,RST04 ;RESTORE REG'S
2414 014620 000002 RTI ;RETURN FROM TRAP CALL
    
```

```

2415 ;SUBROUTINE TO GET PAGE ADDRESS FIELD VALUE (TOP 12 BITS OF ADDRESS)
2416 ;CALL VIA:
2417 ; JSR R5,TOP12S
2418 ; ADDRESS OF TABLE CONTAINING LOWER 16 BITS, TOP 2 BITS (SHIFTED),
2419 ; AND SLOT FOR RETURNED VALUE OF ADDRESS
2420
2421 014622 010046 TOP12S: MOV R0,-(SP) ;SAVE THE OLD VALUE OF R0
2422 014624 012500 MOV (R0)+,R0 ;GET THE ADDRESS OF THE TABLE-PUT IT IN R0
2423 014626 012046 MOV (R0)+,-(SP) ;FETCH THE ADDRESS TO BE ACTED ON TO THE STACK
2424 014630 006216 ASR (SP) ;ELIMINATE THE SIX LOW ORDER BITS. THIS ONE DROPS BIT0
2425 014632 006216 ASR (SP) ;THIS ROTATION DROPS BIT1
2426 014634 006216 ASR (SP) ;THIS ROTATION DROPS BIT2
2427 014636 006216 ASR (SP) ;THIS ROTATION DROPS BIT3
2428 014640 006216 ASR (SP) ;THIS ROTATION DROPS BIT 4
2429 014642 006216 ASR (SP) ;AND THIS ROTATION DROPS BITS
2430 014644 042716 176000 BIC #176000,(SP) ;ISOLATE THE 12 HIGH ORDER BITS IN THE LOWER
2431 014650 011660 000002 MOV (SP),2(R0) ;SAVE THE 12 HIGH ORDER BITS IN THE TABLE'S 3RD ENTRY
2432 014654 012016 MOV (R0)+,(SP) ;FETCH THE EXTENDED ADDRESS BITS FROM THE 2ND ENTRY
2433 014656 000316 SWAB (SP) ;MOVE THE BITS TO BIT POSITIONS 6+7,FIRST. TO 8+9,
2434 014660 006216 ASR (SP) ;THEN MOVE THEM TO 7 + 8
2435 014662 006216 ASR (SP) ;FINALLY ROTATE THEM TO 6 + 7
2436 014664 052610 BIS (SP)+,(R0) ;ADD THE EXTENDED ADDRESS BITS TO THE 3RD ENTRY
2437 014666 012600 MOV (SP)+,R0 ;RESTORE THE OLD VALUE OF R0
2438 014670 000205 RTS R5 ;RETURN TO THE CALLING PROCEDURE
    
```

```

2439 ;ROUTINE TO GET THE 18 BIT PA FROM THE 12 BIT PAF
2440 ;CALL VIA
2441 ; JSR R5,TO18S
2442 ; ADDRESS OF 12 BITS
2443 ; DEST ADDRESS OF LOW 16 BITS
2444 ; DEST ADDRESS OF SHIFTED EA BITS
2445
2446 014672 010046 TO18S: MOV R0,-(SP)
2447 014674 010146 MOV R1,-(SP)
2448 014676 005001 CLR R1
2449 014700 013546 MOV #6,R0
2450 014702 012700 000006 MOV #6,R0
2451 014706 006316 18: ABL (SP)
2452 014710 006101 ROL R1
2453 014712 005300 DEC R0
2454 014714 001374 BNE 18
2455 014716 006301 ABL R1
2456 014720 006301 ASL R1
2457 014722 006301 ASL R1
2458 014724 006301 ASL R1
2459 014726 012635 MOV (SP)+,#(R5)+
2460 014730 010135 MOV R1,#(R5)+
2461 014732 012601 MOV (SP)+,R1
2462 014734 012600 MOV (SP)+,R0
2463 014736 000205 RTS R5
2464

```

```

2465 ;SAVE REGS 0-4 SUBROUTINE. (CALL VIA JSR P4,SAV04)
2466
2467 014740 010346 SAV04: MOV R3,-(SP) ;SAVE R3
2468 014742 010246 MOV R2,-(SP) ;SAVE R2
2469 014744 010146 MOV R1,-(SP) ;SAVE R1
2470 014746 010046 MOV R0,-(SP) ;SAVE R0
2471 014750 010407 MOV R4,PC ;R4 IS ALREADY SAVED
2472
2473 ;RESTORE REGS 0-4 SUBROUTINE. (CALL VIA JSR PC,RST04)
2474
2475 014752 012604 RST04: MOV (SP)+,R4 ;RETURN ADDRESS
2476 014754 012600 MOV (SP)+,R0 ;RESTORE R0
2477 014756 012601 MOV (SP)+,R1 ;R1
2478 014760 012602 MOV (SP)+,R2 ;R2
2479 014762 012603 MOV (SP)+,R3
2480 014764 000204 RTS R4 ;RESTORE R4 AND RETURN

```

```

2481                                     ;CLEAR VARIABLES AND QUEUES, AND LOAD TRAP CATCHER STARTING AT 70
2482
2483 014766 012700 001006 CLRQUS; MOV #IOQUE,R0 ;CLEAR VARIABLES.
2484 014772 012701 000154 MOV #TKS-IOQUE,R1
2485 014776 105020 18; CLR B (R0)+
2486 015000 005301 DEC R1
2487 015002 001375 BNE 18
2488 015004 012700 027252 MOV #IOQ,R0 ;INITIALIZE QUEUE POINTERS
2489 015010 010067 164060 MOV R0,IOQ1
2490 015014 010067 164056 MOV R0,IOQ2
2491 015020 012767 030072 164052 MOV #TYPEQ,TYPQ1
2492 015026 012767 030072 164046 MOV #TYPEQ,TYPQ2
2493 015034 012701 001750 MOV #IOQL+TYPQL,R1 ;SET UP TO CLEAR QUEUES.
2494 015040 105020 28; CLR B (R0)+
2495 015042 005301 DEC R1 ;DONE?
2496 015044 001375 BNE 28 ;BR IF NOT.
2497 015046 000207 RTS PC
    
```

```

2498                                     ;LOAD TRAPCATCHER IN VECTOR AREA
2499
2500 015050 013746 000200 CLRVEC; MOV ##200,-(SP)
2501 015054 013746 000202 MOV ##202,-(SP)
2502 015060 012700 000072 MOV #72,R0 ;FILL VECTOR AREA WITH ,+2
2503 015064 012701 000070 MOV #70,R1 ;AND HALT.
2504 015070 010021 18; MOV R0,(R1)+
2505 015072 005021 CLR (R1)+
2506 015074 010100 MOV R1,R0
2507 015076 005720 TST (R0)+
2508 015100 020027 001002 CMP R0,#1002 ;FILLED UP?
2509 015104 001371 BNE 18 ;BR IF NOT.
2510 015106 012767 015672 163000 MOV #PARERR,PARVCT ;RESTORE PARITY VECTOR
2511 015114 012767 000340 162774 MOV #PRTY7,PARVCT+2 ;AND THE PRIORITY
2512 015122 012637 000202 MOV (SP)+,#202
2513 015126 012637 000200 MOV (SP)+,#200 ;PRESERVE LP VECTOR
2514 015132 000207 RTS PC ;YES, EXIT.
2515
2516                                     ;TRAP INTERPRETER ROUTINE.
2517
2518 015134 010046 TRPINT; MOV R0,-(SP) ;PUSH R0.
2519 015136 016600 000002 MOV 2(SP),R0 ;GET TRAP PC.
2520 015142 014000 MOV -(R0),R0 ;GET TRAP CALL.
2521 015144 006300 ASL R0 ;MULTIPLY BY 2.
2522 015146 062700 004176 ADD #TRPTAB-TRP2,R0 ;FORM TABLE ADDRESS
2523 015152 020027 015246 CMP R0,#TRPLIM ;WITHIN LIMITS?
2524 015156 103002 RHIS 18 ;BR IF NOT.
2525 015160 011000 MOV @R0,R0 ;GET ROUTINE ADDRESS
2526 015162 000200 RTS R0 ;GO TO ROUTINE, RESTORE R0.
2527 015164 104406 023040 18; MSGS,INVT RP ;ERROR INVALID TRAP CALL. CRASH SYSTEM.
2528 015170 104406 023746 MSG6,HLT ;'HALT'
2529 015174 000000 HALT
2530 011000 TRP2=11000
2531 000000 TRAPX=0
2532 015176 TRPTAB;
2533 015176 TRPDEF EXITS,EXIT.
2534 015176 005736 EXIT. ;POINTER FOR TRAP CALL EXITS
2535 015200 TRPDEF QUES,QUE.
2536 015200 006376 QUE. ;POINTER FOR TRAP CALL QUES
2537 015202 TRPDEF ENDPSS,TYPQ.
2538 015202 006232 TYPQ. ;POINTER FOR TRAP CALL ENDPSS
2539 015204 TRPDEF ENDS,TYPQ.
2540 015204 006232 TYPQ. ;POINTER FOR TRAP CALL ENDS
2541 015206 TRPDEF ERRORS,TYPQ2.
2542 015206 006156 TYPQ2. ;POINTER FOR TRAP CALL ERRORS
2543 015210 TRPDEF DATER6,TYPQ2.
2544 015210 006156 TYPQ2. ;POINTER FOR TRAP CALL DATER6
2545 015212 TRPDEF MSG6,TYPQ1.
2546 015212 006244 TYPQ1. ;POINTER FOR TRAP CALL MSG6
2547 015214 TRPDEF BREAK6,LDBRK.
2548 015214 006366 LDBRK. ;POINTER FOR TRAP CALL BREAKS
2549 015216 TRPDEF ERRN6,TYPQ2.
2550 015216 006156 TYPQ2. ;POINTER FOR TRAP CALL ERRN6
2551 015220 TRPDEF MSGN6,TYPQ.
2552 015220 006232 TYPQ. ;POINTER FOR TRAP CALL MSGN6
2553 015222 TRPDEF GWBUF6,GWBUF.
    
```

```

2554 015222 021150          GMBUF,          ; POINTER FOR TRAP CALL GMBUF$
2555 015224          TRPDEF GETPA$,GETPA,
2556 015224 021272          GETPA,          ; POINTER FOR TRAP CALL GETPA$
2557 015226          TRPDEF CDATA$,CDATA,
2558 015226 007636          CDATA,          ; POINTER FOR TRAP CALL CDATA$
2559 015230          TRPDEF MAP22$,MAP22,
2560 015230 021426          MAP22,          ; POINTER FOR TRAP CALL MAP22$
2561 015232          TRPDEF MSGS$,TYPQ,
2562 015232 006232          TYPQ,          ; POINTER FOR TRAP CALL MSGS$
2563 015234          TRPDEF DATCK$,DATCK,
2564 015234 007626          DATCK,          ; POINTER FOR TRAP CALL DATCK$
2565 015236          TRPDEF RAND$,RAND,
2566 015236 014514          RAND,          ; POINTER FOR TRAP CALL RAND$
2567 015240          TRPDEF DERRX$,TYPQ3,
2568 015240 006056          TYPQ3,         ; POINTER FOR TRAP CALL DERRX$
2569 015242          TRPDEF MSGD$,TYPQ4,
2570 015242 006010          TYPQ4,         ; POINTER FOR TRAP CALL MSGD$
2571 015244          TRPDEF RTTK$,RTTK,
2572 015244 014504          RTTK,          ; POINTER FOR TRAP CALL RTTK$
2573 015246          TRPLIM;
    
```

```

2574          ;BUS ERROR AND RESERVED INSTRUCTION TRAP ROUTINES
2575          ;CHECK MODE BITS OF PSW ON STACK (MAYBE USER IF KT11 IN USE)
2576          ;ORDER OF ARGUMENTS: VIRTUAL PC ON STACK, PSW ON STACK, STACK POINTER
2577          ;AFTER TRAP, VIRTUAL ADDRESS TRAPPED TO
2578
2579 015246 012746 000004      BUSERR: MOV     #4,-(SP)          ;INDICATE BUS ERROR TRAP,
2580 015252 000413          BR      RESIA
2581 015254 042737 000001 177572 KTERR: BIC     #1,0*SSRO      ;TURN OFF KT11
2582 015262 104406 024050          MSG$,KTFATL      ;'KT11 ABORT - FATAL'
2583 015266 104406 023746          MSG$,HLT        ;'HALT'
2584 015272 000000          HALT
2585 015274 000776          BR      #-2
2586          ;KT11 ABORT- FATAL ERROR
2587          ;EXAMINE KT11 SSRO TO DETERMINE TYPE OF ABORT,
2588          ;MODE, AND PAGE REFERENCED
2589          ;EXAMINE KT11 SSP2 TO DETERMINE VIRTUAL ADDRESS
2590          ;OF INSTRUCTION WHICH ABORTED
2591          ;INDICATE RESERVED INSTRUCTION TRAP,
2592
2593 015276 012746 000010      RESINT: MOV     #10,-(SP)
2594 015302 000240          RESIA: NOP
2595 015304 010603          MOV     SP,R5
2596 015306 012706 027252          MOV     #SPSPEC,SP      ;SAVE THE STACK POINTER,
2597 015312 012546          MOV     (R5)+,-(SP)      ;PUT INFORMATION INTO SPECIAL MONITOR STACK
2598 015314 010546          MOV     R5,-(SP)        ;PUSH TRAP VECTOR ON STACK
2599 015316 016546 000002          MOV     2(R5),-(SP)     ;PUSH STACK POINTER ON STACK
2600 015322 011546          MOV     (R5),-(SP)     ;PUSH THE PSW ON STACK
2601 015324 012706 027212          MOV     #SPBOT,SP      ;PUSH THE PC ON STACK
2602 015330 004767 002152          JSR     PC,SAVLOG      ;RESTORE MONITOR STACK POINTER
2603 015334 004767 002774          JSR     PC,CLKOFF      ;GO SAVE 11/60 LOG *****
2604 015340 004767 171640          JSR     PC,CLKOFF      ;TURN OFF CLOCK AND SAVE STATUS
2605 015344 105267 163714          JSR     PC,CTRLX      ;CLEAR QUEUES, CLEAN UP, MAP TO BANKS 0+1
2606 015350 105767 163751          INCB   SYSCNT
2607 015354 001005          TSTR   MOVING
2608 015356 105767 163722          BNE    1$
2609 015362 001402          TSTR   KTRPRES
2610 015364 004767 004104          BEQ    1$
2611 015370 012706 027242          JSR     PC,MVCODE      ;MAP BACK TO CURRENT COPY
2612 015374 005067 163460          MOV     #SPSPEC-10,SP  ;SET UP THE SPECIAL STACK POINTER
2613 015400 112767 000001 163405          CLR    IORXID
2614 015406 104406 023601          MOV     #1,BRAKE
2615 015412 012705 013426          MSG$,SYSERR
2616 015416 004715          MOV     #ITOA,R5
2617 015420 004715          JSR     PC,(R5)
2618 015422 004715          JSR     PC,(R5)
2619 015424 004715          JSR     PC,(R5)
2620 015426 004715          JSR     PC,(R5)
2621 015428 004715          JSR     PC,(R5)
2622 015432 004715          JSR     PC,(R5)
2623 015434 004715          JSR     PC,(R5)
2624 015436 000435          BR      PWRUA
    
```

```

2625          ;POWER DOWN AND POWER UP ROUTINES.
2626
2627 015460 004767 002650      PWRDN: JSR      PC,CLKOFF      ;TURN OFF CLOCK AND SAVE STATUS
2628 015464 105767 163614      TSTB     KTPRES      ;KT11 IN USE ?
2629 015470 001402              BEQ      18          ;NO, CONTINUE
2630 015472 005037 177572      CLR      #SSRO      ;YES, TURN IT OFF
2631 015476 012767 015506 162320 18:  MOV      #PWRUP,PWRFV ;SET UP POWER UP VECTOR.
2632 015504 000000              HALT                    ;POWER DOWN HALT
2633 015506 012767 015460 162310 PWRUP: MOV      #PWRDN,PWRFV ;SET UP POWER DOWN VECTOR.
2634 015514 012706 027212      MOV      #SPBOT,R6  ;RESET STACK.
2635 015520 005267 163542      INC      PWRCNT     ;COUNT A POWER FAIL
2636 015524 105067 163576      CLR     RUNRFL     ;CLEARRESUME RUN FLAG
2637 015530 004767 171450      JSR     PC,CTRLX   ;CLEAR QUEUES AND INITIALIZE KT11 IF IN USE.
2638 015534 005067 163320      CLR     IOBKID    ;ALLOW TYPING
2639 015540 112767 000001 163245  MOV     #1,BRAKE   ;PREVENT MODULE STARTUP
2640 015546 104406 023702      MSGS,PWRFAI      ;TYPE POWER FAILURE MESSAGE.
2641
2642 015552              PWRUA: TSTB     KTPRES      ;KT11 IN USE ?
2643 015556 001412              BEQ      18          ;NO, CONTINUE
2644 015560 042737 000001 177572  BIC     #1,SSRO    ;YES, BACK TO LOW BANK
2645 015566 116701 163472      MOV     SYSCNT,R1  ;SAVE THE SYSTEM ERROR COUNT
2646 015572 052737 000001 177572  BIS     #1,SSRO    ;BACK TO HIGHER BANK
2647 015600 110167 163460      MOV     R1,SYSCNT ;PUT SYS ERR COUNT IN THIS MEMORY BANK
2648 015604 126767 163540 163452 18:  CMPB   SYSLIM,SYSCNT ;REACHED THE SYS. ERR. LIMIT ?
2649 015612 101004              BHI     28          ;NO, CONTINUE
2650 015614 104406 024145      MSGS,SYSBAD     ;"EXERCISE ABORTED"
2651 015620 000167 171220      JMP     CTRLCA   ;CTRL C AND KILL THE EXERCISE
2652 015624 132767 000002 163506 28:  BITB   #BIT1,PPRES ;IS THIS AN 11/60 ? *****
2653 015632 001413              BEQ     38          ;NO, CONTINUE
2654 015634 012767 100001 163632  MOV     #100001,LGHOLD ;YES, LOAD ERROR LOG ENABLED BIT
2655 015642              WRITE  RDWHMI,WTWHMI ;WRITE WHAMI REGISTER
2656 015644 010046              MOV     RO,-(SP)   ;SAVE REG. 0
2657 015644 076600 000022      MED,RDWHMI     ;SAVE
2658 015650 056700 163620      BIS     LGHOLD,R0
2659 015654 076600 000222      MED,WTWHMI
2660 015660 012600              MOV     (SP)+,RO  ;RESTORE REG. 0
2661 015662 105767 163423 38:  TSTB   RMODE     ;WERE WE IN RUN MODE?
2662 015666 001126              BNE     LOGICA    ;BR IF YES TO RESTART.
2663 015670 000505              BR      CHNOUT    ;NO, CONTINUE
    
```

```

2664          ;PARITY ERROR TRAP HANDLER ROUTINE
2665
2666 015672 022626      PARERR: CMP     (SP)+,(SP)+ ;POP THE STACK POINTER TWICE
2667 015674 004767 002434      JSR     PC,CLKOFF ;MAKE SURE SYSTEM CLOCK IS OFF
2668 015700 132767 000001 163432  BITB   #BIT0,PPRES ;IS PARITY PRESENT/ENABLED ?
2669 015706 001002              BNE     28          ;YES, CONTINUE
2670 015710 000000 18:  HALT                    ;NO PARITY BUT TRAPPED ANYWAY
2671
2672 015712 000776      BR      18          ;RUN PARITY DIAGNOSTICS
2673
2674 015714 132767 000002 163416 28:  BITB   #BIT1,PPRES ;IS IT AN 11/60 ? *****
2675 015722 001405              BEQ     38          ;NO, CONTINUE
2676 015724 104406 025427      MSGS,PERR2     ; "PARITY ERROR"
2677 015730 004767 000076      JSP     PC,PAMSG1 ;GO FINISH PARITY ERROR MESSAGE
2678 015734 000706      BR      PWRUA     ;ATTEMPT TO KEEP GOING
2679
2680 015736 012700 001430 38:  MOV     #PARTAB,R0 ;GET ADDRESS OF PARITY ADDRESSES
2681 015742 005001      CLR     R1          ;ZERO THE REGISTER COUNTER
2682 015744 005770 000000 48:  TST     @R0        ;ERROR BIT SET ?
2683 015750 100407      BMI     68          ;YES, GO REPORT IT
2684 015752 005720      TST     (R0)+     ;STEP TO THE NEXT REGISTER ADDRESS
2685 015754 005201      INC     R1        ;STEP THE REGISTER COUNTER
2686 015756 126701 163357      CMPB   CNT1,R1   ;ALL REGISTERS CHECKED ?
2687 015762 003365      BGT     38          ;NO, CONTINUE
2688 015764 000000 58:  HALT                    ;YES, TRAPPED BUT NO ERRORS SET
2689
2690 015766 000776      BR      58          ;RUN PARITY DIAGNOSTICS
2691
2692 015770 004767 000002 68:  JSR     PC,PAMSG ;GO PRINT PARITY ERROR MESSAGE
2693 015774 000666      BR      PWRUA     ;ATTEMPT TO KEEP GOING
    
```

```

2694 ;PARITY ERROR MESSAGE ROUTINE, ASSUMES THAT THE ADDRESS OF THE FAILING
2695 ; PARITY REGISTER IS IN R0.
2696
2697 PAMSG: MOV (R0),-(SP) ;GET ADDRESS OF FAILING PARITY REG.
2698 MOV #60,=(SP) ;LOAD THE EA BITS
2699 JSR R5,OACNVX ;CONVERT IT TO OCTAL ASCII
2700 PREG ;PUT IT INTO LOC. PREG
2701 MOV #*(R0),=(SP) ;GET CONTENTS OF FAILING PARITY REG.
2702 JSR R5,OACNV ;CONVERT IT TO OCTAL ASCII
2703 PCONT ;PUT IT INTO LOC. PCONT
2704 MSG$,PERR1 ;"PARITY ERROR -- RUN PARITY DIAG."
2705 RTS PC ;RETURN
2706
2707
2708 PAMSG1: JSR PC,SAVLOG ;SAVE ERROR LOG IN BUFFER *****
2709 MOV #LOGBUF+200,R2 ;GET ADR. OF SERVICE REGISTER (100*2)
2710 MOV #10,R1 ;SET THE COUNTER
2711 MOV (R2)+,=(SP) ;PUT CONTENTS OF LOG REG. ON STACK
2712 JSR R5,OACNV ;CONVERT IT TO OCTAL
2713 HOLD6 ;AND PUT IT HERE
2714 MOV #HOLD6,=(SP) ;PUT IT ON THE STACK
2715 JSR PC,TYPE ;AND TYPE IT
2716 DEC R1 ;ALL DONE ?
2717 BNE 1$ ;NO, CONTINUE
2718 MOV #ACRLF,=(SP) ;PUT A CR AND LF ON THE STACK
2719 JSR PC,TYPE ;AND TYPE IT
2720 RTS PC ;RETURN
2721

```

```

2722 ;ROUTINE TO EXIT TO CHAIN MONITOR, OR RETURN TO KYBD RTN.
2723 ;PROGRAM SHOULD ALWAYS BE IN BANK 0 BY HERE
2724
2725 CHNOUT: TSTB CHN ;IN CHAIN MODE?
2726 BMI 1$ ;BR IF YES.
2727 JMP COMCON ;BACK TO KYBD SERVICE.
2728 1$: TSTB KTPRES
2729 BEQ 2$ ;BRANCH IF NO KT11
2730 CLR #SSRO ;TURN OFF KT11
2731 MOV 42,R0
2732 2$: JSR PC,(R0) ;RETURN TO MONITOR.
2733 LOGIC: WORD NOP,NOP,NOP
2734 LOGICA: TSTB CHN ;IN CHAIN MODE?
2735 BPL 3$ ;BR IF NO
2736 TSTB KTSAV ;YES, KT REALLY PRESENT?
2737 BEQ 4$ ;BR IF NOT
2738 MOV HCR12A,HCDR12 ;SET REAL MEMORY SIZE
2739 MOV HCR12A,OFFMX ;TOP ALLOWABLE PROGRAM OFFSET
2740 INCB ROT ;ALLOW WBUFF ROTATION
2741 INCB RELOC ;ALLOW CODE RELOCATION
2742 MOV #1,CHN ;ONLY RETURN TO CHAIN MONITOR ONCE
2743 MOV KTSAV,KTPRES ;TELL THE TRUTH
2744 3$: TSTB KTPRES
2745 BEQ 5$
2746 BIT #1,#SSRO ;KT11 ALREADY ON?
2747 BNE 2$ ;YES-SKIP REINITIALIZING
2748 JSR PC,KTIWIT
2749 MOV #1,#SSRO
2750
2751
2752 MOV OFFSET,OFFSTD 162766 2$: MOV
2753 JSR PC,MVCCODE
2754 JSR PC,CLKON ;TURN CLOCK ON
2755 5$: TSTB CHN
2756 BEQ 6$
2757 JMP START 163234
2758 6$: JMP RUNRES 172630

```

```

2759 ;SUBROUTINE TO ROTATE WRITE BUFFERS IF ROTI INDICATOR #1.
2760
2761 016304 105767 163003 ROTBUF: TSTB ROTI ;ROTATION ALLOWED?
2762 016310 001557 BEQ ROTXT ;BR IF NOT.
2763 016312 105767 162776 TSTR ROTL ;IS ROTATION LOCKED ?
2764 016316 100016 BPL 18 ;NO, CONTINUE
2765 016320 105767 163003 TSTB LOCK ;YES, IS THE PROGRAM LOCKED ?
2766 016324 001013 BNE 18 ;YES, DON'T SET STOP
2767 016326 005367 162764 DEC ROTCNT ;NO, TIME TO RELOCATE ?
2768 016332 003146 BGT ROTXT ;NO, CONTINUE
2769 016334 105767 163012 TSTB NSTOP ;ALLOW STOP TO SET ?
2770 016340 001002 BNE 108 ;NO, CONTINUE
2771 016342 105267 162456 INCB 108 ;YES, SET THE STOP FLAG
2772 016346 016767 162746 MOV ROTNUM,ROTCNT ;RESTORE THE COUNTER
2773 016354 105767 162444 18: TSTB STOP ;IF STOP IS SET, DON'T ROTATE
2774 016360 001133 BNE ROTXT
2775 016362 105767 162726 TSTB ROTL ;ROTATE LOCKED?
2776 016366 100530 BMI ROTXT ;BR IF YES
2777 016370 105767 162425 TSTB MDCXTR ;IF COUNTER NOT INITIALIZED, DON'T ROTATE
2778 016374 100525 BMI ROTXT
2779 016376 016746 162630 MOV WRUF, -(SP) ;CHECK CURRENT VALUE
2780 016402 042716 160000 BIC #16000, (SP)
2781 016406 022726 C16000 CNP #16000, (SP)+
2782 016412 003016 BGT 38 ;IF NOT LAST IN BANK, BRANCH
2783 016414 105767 162401 TSTB MDCXTR ;HAVE ALL MODULES USED A VALUE IN THIS BANK?
2784 016420 001403 BEQ 28 ;YES- TIME TO CHANGE BANKS
2785 016422 105367 162706 DECB #GWBFC ;NO, DEC TIMEOUT COUNTER
2786 016426 001110 BNE ROTXT ;EXIT UNLESS TIMEOUT HAS OCCURRED
2787 016430 116767 162366 162363 28: MOVB MDCXNT, MDCXTR ;RESET COUNTERS
2788 016436 116767 162367 162670 MOVB ROTCT, #GWBFC
2789 016444 105167 162353 COMB #BFLG ;TOGGLE INDICATOR
2790 016450 026767 162564 38: CMP #BUF12, OFFSET ;LOCORE ROTATION?
2791 016456 101453 BLOS ROTL01 ;YES- BRANCH
2792 016460 062767 162550 ADD #20, #WBUF12
2793 016466 026767 162544 162532 CMP #WRUF12, #COR12 ;CHECK FOR TOP OF WRITE BUFFER AREA
2794 016474 103427 BLO ROTB2 ;IF NOT, BRANCH
2795 016476 000400 BR 46 ;IF OVER, START AGAIN
2796 016500 026727 162536 000400 48: CMP OFFSET, #400 ;ROTATE THRU MEMORY BELOW PROGRAM IF ENOUGH ROOM
2797 016506 103403 BLO ROTB1 ;NOT ENOUGH ROOM
2798 016510 004767 000136 JSR PC, ROTLOW ;SETUP TO ROTATE BELOW PROGRAM
2799 016514 000455 BR ROTXT ;RETURN
2800 016516 105767 162573 ROTB1: TSTR RELOC ;RELOCATION ALLOWED?
2801 016522 001426 BEQ ROTB3 ;NO- BRANCH
2802 016524 105767 162577 TSTR LOCK ;LOCKED?
2803 016530 001023 BNE ROTR3 ;YES- BRANCH
2804 016532 105767 162614 TSTB NSTOP ;ALLOW STOP TO SET ?
2805 016536 001002 BNE 18 ;NO, CONTINUE
2806 016540 105267 162260 INCB 18 ;YES, SET STOP
2807 016544 162767 000020 162464 18: SUB #20, #WBUF12 ;RESTORE #BUF
2808 016552 000436 BR ROTXT ;EXIT
2809 016554 062767 002000 162450 ROTB2: ADD #2000, WRUF ;BUMP ADDR- POINT 512 WORDS HIGHER
2810 016562 103003 BCC 18
2811 016564 062767 000020 162442 ADD #20, #WFPEA
2812 016572 004767 173130 18: JSR PC, SETWB1 ;CALCULATE #X#BF
2813 016576 000424 BR ROTXT
2814 016600 004767 173040 ROTB3: JSR PC, SETWBF
    
```

```

2815 016604 000421 BR ROTXT ;RETURN
2816
2817 016606 062767 000020 162422 ROTL01: ADD #20, #WBUF12
2818 016614 026767 162416 162420 CMP #WRUF12, OFFSET
2819 016622 103401 RLO ROTR4 ;OK- USE IT
2820 016624 000734 RR ROTR1 ;NO- RELOCATE OR START #BUF CYCLE OVER
2821 016626 062767 002000 162376 ROTB4: ADD #2000, WRUF ;ADVANCE THE WRITE BUFFER
2822 016634 103003 BFC 18 ;IF NO CARRY DON'T ADJUST THE EA BITS
2823 016636 062767 000020 162370 ADD #20, #WFPEA ;CARRY SET, ADJUST THE EA BITS
2824 016644 004767 000016 18: JSR PC, ROTLW1 ;TAKE CARE OF ROTATING BELOW THE PROGRAM
2825 016650 000207 ROTXT: RTS ;RETURN
    
```

```

2826 ;THIS ROUTINE WILL HANDLE THE SITUATION WHEN THE WRITE BUFFER IS ROTATING
2827 ; BENEITH THE PROGRAM
2828
2829 016652 005067 162354 ROTLW: CLR WBUF
2830 016656 005067 162352 CLR WBFEA
2831 016662 005067 162350 CLR WBUF12
2832 016666 016767 162350 162362 ROTLW: MOV OFFSET,MXWBF
2833 016674 166767 162336 162354 SUB WBUF12,MXWBF
2834 016702 006367 162350 ASL MXWBF
2835 016706 006367 162344 ASL MXWBF
2836 016712 006367 162340 ASL MXWBF
2837 016716 006367 162334 ASL MXWBF
2838 016722 006367 162330 ASL MXWBF
2839 016726 103003 BCC 1$
2840 016730 012767 177777 162320 MOV #177777,MXWBF
2841 016736 000207 16: RTS PC ;RETURN
2842
2843
2844 ;SUBROUTINE TO CHECK IF BUFFER ROTATION ENABLED, AND TO TYPE
2845 ;THE ROTATION RANGE.
2846
2847 016740 105767 162371 ROTCHK: TSTB MEMERF ;ANY MEMORY ERRORS?
2848 016744 001406 BEQ 1$ ;NO- BRANCH
2849 016746 104406 026344 MSGS,MERR1
2850 016752 104406 023724 MSGS,MEMCHG
2851 016756 105067 162353 CLRB MEMERF
2852 016762 105767 162325 16: TSTB ROTI ;ROTATE ENABLED?
2853 016766 001430 BEQ 2$ ;BR IF NOT.
2854 016770 104406 024505 MSGS,ROTEB
2855 016774 016746 162220 MOV LOCORE,=(SP) ;TYPE LOWER LIMIT.
2856 017000 004767 174422 JSR PC,ITOA
2857 017004 004567 175662 JSR R5,TO18S ;CONVERT UPPER LIMIT
2858 017010 001226 HCOR12
2859 017012 001132 SAVLO
2860 017014 001134 SAVHI
2861 017016 162767 000002 162106 SUB #2,SAVLO
2862 017024 103003 BCC 3$
2863 017026 162767 000020 162100 SUB #20,SAVHI
2864 017034 016746 162072 36: MOV SAVLO,=(SP) ;TYPE UPPER LIMIT
2865 017040 016746 162070 MOV SAVHI,=(SP)
2866 017044 004767 174266 JSR PC,ITOA
2867
2868 017050 000207 28: RTS PC ;EXIT.
    
```

```

2869 ;THIS ROUTINE PRINTS OUT THE SYSTEM MEMORY SIZE IN DECIMAL.
2870
2871 017052 016700 162150 SYSIZ: MOV HCOR12,R0 ;GET THE MEMORY SIZE
2872 017056 006200 ASR R0 ;CONVERT FROM BYTE SIZE TO K WORD SIZE
2873 017060 006200 ASR R0 ;
2874 017062 006200 ASR R0 ;
2875 017064 006200 ASR R0 ;
2876 017066 006200 ASR R0 ;
2877 017070 010046 MOV R0,=(SP) ;PUT IT ON THE STACK
2878 017072 004567 175274 JSR R5,RDCNV ;CONVERT TO DECIMAL ASCII CHARACTERS
2879 017076 025667 HOLD5 ;AND PUT THEM HERE
2880 017100 012700 025667 MOV #HOLD5,R0 ;GET THE LOCATION OF THE ASCII NUMBER
2881 017104 122720 000060 16: CMPR #60,(R0)+ ;IS IT A LEADING 0?
2882 017110 001000 BNE 2$ ;NO, CONTINUE
2883 017112 112760 000040 177777 MOVB #40,=(R0) ;YES, FILL IT WITH A BLANK
2884 017120 000771 BR 1$ ;CHECK THE NEXT CHARACTER
2885 017122 104406 024465 MSGS,SYSIZ ;"SYSTEM SIZE:"
2886 017126 104406 025667 MSGS,HOLD5 ;TYPE THE SYSTEM SIZE
2887 017132 104406 024503 MSGS,KAY ;"K"
2888 017136 000207 RTS PC ;RETURN
    
```

```
2889          ; THIS ROUTINE WILL PRINT THE NUMBER OF SYSTEM ERRORS
2890          ; FOUND IN COUNTER SYSCNT. CALL VIA: JSR PC,SYSDUM
2891
2892 017140 010046 SYSDUM: MOV R0,-(SP) ;SAVE REG. 0
2893 017142 010146      MOV R1,-(SP) ;SAVE REG. 1
2894 017144 016746 162114      MOV SYSCNT,-(SP) ;PUT SYSTEM ERROR COUNT ON STACK
2895 017150 004567 175216      JSR RS,BDCNV ;CONVERT IT TO DECIMAL ASCII CHARS.
2896 017154 025667      HOLDS ;AND PUT THEM HERE
2897 017156 012701 000004      MOV #4,R1 ;LOAD THE COUNTER
```

```
2898 017162 012700 025667      MOV #HOLDS,P0 ;GET ADR OF THE ASCII NUMBER
2899
2900 017166 122720 000060      10: CMPB #6,(R0)+ ;IS IT A LEADING ZERO ?
2901 017172 001005      BNE 20 ;NO, CONTINUE
2902 017174 112760 000040 177777      MOVB #40,-1(R0) ;YES, REPLACE IT WITH A BLANK
2903 017202 005301      DEC R1 ;ALL DONE ?
2904
2905 017204 001370      BNE 10 ;NO, CHECK MORE CHAPACTERS
2906
2907 017206 104406 025531      20: MSG#,NUMSYS ;"SYSTEM ERROPS:"
2908 017212 104406 025667      MSG#,HOLDS ;TYPE THE NUMBER
2909
2910 017216 012601      MOV (SP)+,R1 ;RESTORE REG. 1
2911 017220 012600      MOV (SP)+,R0 ;RESTORE REG. 0
2912 017222 000207      RTS PC ;RETURN
```

```

2913 ; THIS ROUTINE WILL PRINT THE NUMBER OF POWER FAIL ERRORS
2914 ; FOUND IN COUNTER PWRCNT. CALL VIA: JSR PC,PWRNUM
2915
2916 PWRNUM: MOV R0,-(SP) ;SAVE REG. 0
2917 MOV R1,-(SP) ;SAVE REG. 1
2918 MOV PWRCNT,=(SP) ;PUT POWER FAIL ERROR COUNT ON STACK
2919 JSR R5,BDCNV ;CONVERT IT TO DECIMAL ASCII CHARS.
2920 HOLDS ;AND PUT THEM HERE
2921 MOV #4,R1 ;LOAD THE COUNTER
2922 MOV #HOLDS,R0 ;GET ADR OF THE ASCII NUMBER
2923
2924 18: CMPB #60,(R0)+ ;IS IT A LEADING ZERO ?
2925 BNE 28 ;NO, CONTINUE
2926 MOVB #40,-1(R0) ;YES, REPLACE IT WITH A BLANK
2927 DEC R1 ;ALL DONE ?
2928 BNE 18 ;NO, CHECK MORE CHARACTERS
2929
2930 28: MSG#,NUMPWR ;"POWER FAILS:"
2931 MSG#,HOLDS ;TYPE THE NUMBER
2932
2933 MOV (SP)+,R1 ;RESTORE REG. 1
2934 MOV (SP)+,R0 ;RESTORE REG. 0
2935 RTS PC ;RETURN
2936

```

```

2937 ; THIS ROUTINE WILL CONVERT THE GIVEN BINARY
2938 ; NUMBER INTO HOURS, MINUTES, AND SECONDS. LOCATIONS
2939 ; HOLDING THE FINAL NUMBERS WILL BE TAGGED AS: HOURS,
2940 ; MINITS, AND SEKND. HOURS MUST BE ON AN ODD BOUNDARY.
2941 ; MINITS AND SEKND. MUST BE ON EVEN BOUNDARIES.
2942 ; TEMPORARY LOCATION HOLDS MUST BE ON AN ODD BOUNDARY.
2943 ; CALL VIA: JSR PC,HMS
2944 ;
2945 ;
2946 ;
2947 HMS: JSR R4,SAV04 ;SAVE REGS. 0-4
2948 MOV 10,(SP),R0 ;GET ADR OF OF ADR TIME
2949 MOV R0,R3 ;COPY IT INTO REG. 3
2950 TST #R3+ ;GET ADR OF EXTENDED TIME BITS ADR
2951 MOV #R0,R0 ;PUT THE TIME IN REG. 0
2952 MOV #R3,R3 ;PUT EXTENDED TIME BITS INTO REG. 3
2953 CLR R1 ;ZERO MINUTES COUNTER
2954 CLR R2 ;ZERO HOURS COUNTER
2955
2956 18: CMP #3600,,R0 ;LESS THAN 1 HOUR?
2957 BHI 38 ;YES, GO LOOK FOR MORE HOURS
2958 SUB #3600,,R0 ;NO, SUBTRACT 1 HOUR'S WORTH OF SECONDS
2959 INC R2 ;COUNT AN HOUR
2960 BR 18 ;CHECK FOR MORE HOURS
2961
2962 28: CMP #60,,R0 ;LESS THAN 1 MINUTE?
2963 BHI 48 ;YES, ALL DONE
2964 SUB #60,,R0 ;NO, SUBTRACT 1 MINUTE'S WORTH OF SECONDS
2965 INC R1 ;COUNT A MINUTE
2966 BR 28 ;CHECK FOR MORE MINUTES
2967
2968 38: TST R3 ;ANY EXTENDED TIME BITS ?
2969 BEQ 48 ;NO, GO LOOK FOR MINUTES
2970 ADD #17,,R2 ;YES, ADD IN 17. HOURS
2971 ADD #4336,,R0 ;ADD IN 1 HR 12 MIN 16 SEC
2972 DEC R3 ;GET RID OF ONE TIME EXTENDED BIT
2973 BR 18 ;CONTINUE
2974
2975 48: MOV R2,-(SP) ;PUT THE HOURS ON THE STACK
2976 JSR R5,BDCNV ;CONVERT IT TO ASCII
2977 HOLDS ;AND PUT IT HERE
2978 MOVB HOLDS+2,HOURS ;LOAD HOURS
2979 MOV HOLDS+3,HOURS+1 ;LOAD HOURS
2980 MOV R1,-(SP) ;PUT THE MINUTES ON THE STACK
2981 JSR R5,BDCNV ;CONVERT IT TO ASCII
2982 HOLDS ;AND PUT IT HERE
2983 MOV HOLDS+3,MINITS ;LOAD MINUTES
2984 MOV R0,-(SP) ;PUT THE SECONDS ON THE STACK
2985 JSR R5,BDCNV ;CONVERT IT TO ASCII
2986 HOLDS ;AND PUT IT HERE
2987 MOV HOLDS+3,SEKND ;LOAD SECONDS
2988
2989 JSR PC,RST04 ;RESTORE REGS 0-4
2990 ADD #4,(SP) ;SET PROPER RETURN PC
2991 RTS PC ;AND RETURN
2992

```

2993

```
2994          ;          THIS ROUTINE COPIES THE 11/60 ERROR LOG AND PUTS IT INTO
2995          ;A BUFFER IN THE MONITOR.
2996
2997 017506 132767 000002 161624 SAVLOG: BITB  #BIT1,PPRES ;IS IT AN 11/60 ? *****
2998 017514 001414          BEQ      3$          ;NO, GET OUT
2999 017516 012701 031422          MOV      #LOGBUF,P1 ;GET ADDR. OF LOG BUFFER
3000 017522 005002          CLR      R2          ;INITIALIZE LOG COMMAND AND ADDRESS
3001
3002 017524 010267 000002 18:    MOV      R2,2$          ;LOAD READ COMMAND AND THE ADDRESS
3003 017530 076600          MED          ;READ THE LOG REGISTER
3004 017532 000000 25:    ,WORD          ;READ COMMAND AND REGISTER ADDRESS
3005 017534 010021          MOV      R0,(P1)+ ;STORE CONTENTS OF LOG IN BUFFER
3006 017536 005202          INC      R2          ;STEP TO NEXT ADDRESS
3007 017540 022702 000177          CMP      #177,P2 ;ALL DONE ?
3008 017544 001367          BNE     1$          ;NO, KEEP GOING
3009
3010 017546 000207 35:    RTS      PC          ;YES, RETURN
3011
```

```

3012 ; THIS ROUTINE CHECKS TO SEE IF ANY CLOCK
3013 ;MODULES ARE PRESENT AND SETS BIT FLAGS IN BYTE
3014 ;LOCATION CLOCK
3015 ; CALL VIA: JSR PC,KLOCK
3016 ;
3017
3018 017550 105067 161577 KLOCK: CLR B CLOCK ;ZERO CLOCK FLAG
3019 017554 005767 161576 TST CLOCKL ;KW11-L CLOCK PRESENT?
3020 017560 001406 BEQ 18 ;NO, CONTINUE
3021 017562 112767 000001 161563 MOVB #BIT0,CLOCK ;SET L CLOCK PRESENT FLAG
3022 017570 017700 161562 MOV @CLOCKL,R0 ;GET BEGIN ADDRESS OF CLOCK MODULE
3023 017574 000410 BR 28 ;CONTINUE
3024
3025 017576 005767 161556 18: TST CLOCKP ;KW11-P CLOCK PRESENT?
3026 017602 001421 BEQ 48 ;NO, RETURN
3027 017604 112767 000002 161541 MOVB #BIT1,CLOCK ;SET P CLOCK PRESENT FLAG
3028 017612 017700 161542 MOV @CLOCKP,R0 ;GET BEGIN ADDRESS OF CLOCK MODULE
3029 017616 012737 000004 28: MOV #36,R#4 ;LOAD TIME-OUT VECTOR
3030 017624 005070 000006 CLR @6(R0) ;MAKE SURE SYSTEM CLOCK IS THERE
3031 017630 000406 BR 48 ;IT'S THERE, RETURN OK
3032
3033 017632 022626 36: CMP (SP)+,(SP)+ ;POP THE STACK POINTER TWICE
3034 017634 104406 024317 MSG@,NOCLK ;"NO SYSTEM CLOCK FOUND"
3035 017640 000000 HALT ;WAIT FOR OPERATOR'S DECISION
3036 017642 105067 161505 CLR B CLOCK ;INDICATE NO CLOCK AVAILABLE
3037 017646 012737 015246 000004 48: MOV #BUSERR,R#4 ;RESTORE TIME-OUT VECTOR
3038 017654 000207 RTS PC ;RETURN
    
```

```

3039 ; THIS ROUTINE WILL GET THE RUNTIME IN
3040 ;SECONDS FROM THE CLOCK MODULE, HAVE IT CONVERTED
3041 ;TO HOURS, MINUTES, AND SECONDS VIA SUBROUTINE HMS, AND
3042 ;STUFFS IT INTO PROPER LOCATION FOR PRINTOUT.
3043 ; CALL VIA: JSR PC,RUNTYM
3044 ;
3045 017656 010046 RUNTYM: MOV R0,-(SP) ;SAVE REG. 0
3046 017660 132767 000001 161465 BITB #BIT0,CLOCK ;USING THE KW11-L CLOCK?
3047 017666 001005 BNE 18 ;YES, CONTINUE
3048 017670 132767 000002 161455 BITB #BIT1,CLOCK ;USING THE KW11-P CLOCK?
3049 017676 001004 BNE 28 ;YES, CONTINUE
3050 017700 000456 BR 58 ;NO, CAN'T DO ANYTHING, RETURN
3051
3052
3053
3054
3055
3056
3057
3058
3059 017702 016700 161450 18: MOV CLOCKL,R0 ;GET ADR OF CLOCK INFORMATION
3060
3061
3062 017706 000402 BR 38 ;CONTINUE
3063 017710 016700 161444 28: MOV CLOCKP,R0 ;GET ADR OF CLOCK INFORMATION
3064
3065 017714 016067 000002 161472 36: MOV 2(R0),PTIME ;GET THE TIME FROM THE CLOCK MODULE
3066 017722 016067 177776 161466 MOV -2(R0),PTIMX ;GET THE TIME EXTENDED BITS TOO
3067 017730 105767 161355 TSTB RMODE ;PROGRAM IN THE RUN MODE ?
3068 017734 001407 BEQ 48 ;NO, MUST BE TIME FOR SUMMARY
3069 017736 105767 161407 TSTB TFLAG ;YES, IS IT ERROR TIME ?
3070 017742 100404 BMI 48 ;YES, SETUP RUNTIME FOR OUTPUT
3071 017744 032777 010000 161224 BIT #BIT12,@SP ;NO, PRINT THE EDP DATA ?
3072
3073
3074 017752 001431 BEQ 58 ;NO, RETURN
3075
3076 017754 004767 177330 48: JSR PC,HMS ;CONVERT TO HOURS, MINUTES AND SECONDS
3077 017760 001414 RTIME ;FROM SECONDS
3078 017762 001416 RTIMX ;EXTENDED BITS
3079 017764 116767 161415 006105 MOVB HOURS,THRS ;LOAD 100'S COLUMN OF HOURS
3080 017772 116767 161410 006100 MOVB HOURS+1,THRS+1 ;LOAD 10'S COLUMN OF HOURS
3081 020000 116767 161403 006073 MOVB HOURS+2,THRS+2 ;LOAD 1'S COLUMN OF HOURS
3082 020006 116767 161376 006067 MOVB MINITS,TMINS ;LOAD 10'S COLUMN OF MINUTES
3083 020014 116767 161371 006062 MOVB MINITS+1,TMINS+1 ;LOAD 1'S COLUMN OF MINUTES
3084 020022 116767 161364 006056 MOVB SEKND$TSECS ;LOAD 10'S COLUMN OF SECONDS
3085 020030 116767 161357 006051 MOVB SEKND$+1,TSECS+1 ;LOAD 1'S COLUMN OF SECONDS
3086
3087 020036 012600 58: MOV (SP)+,R0 ;RESTORE REG. 0
3088 020040 000207 RTS PC ;RETURN
    
```

```

3089
3090
3091
3092
3093
3094
3095
3096
3097 020042 112767 000001 161301 BKTYM; MOVB #BIT0,TFLAG ;SET FLAG INDICATING BKMOD TIME SETUP
3098 020050 000406 BR RUNPAS ;CONTINUE
3099 020052 112767 000200 161271 ERRTYM; MOVB #BIT7,TFLAG ;SET ERROR CALL FLAG
3100 020060 000402 BR RUNPAS ;CONTINUE
3101 020062 105067 161263 PASTYM; CLRB TFLAG ;SET END OF PASS FLAG
3102
3103 020066 004467 174646 RUNPAS; JSR R4,SAV04 ;SAVE REGS. 0-4
3104 020072 132767 000001 161253 BITB #BIT0,CLOCK ;USING THE KW11-L CLOCK?
3105 020100 001005 BNE 10 ;YES, CONTINUE
3106 020102 132767 000002 161243 BITB #BIT1,CLOCK ;USING THE KW11-P CLOCK?
3107 020110 001004 BNE 20 ;YES, CONTINUE
3108 020112 000505 BR 90 ;NO, CAN'T DO ANYTHING, RETURN
3109
3110 020114 016700 161236 10; MOV CLOCKL,R0 ;GET ADR OF CLOCK INFORMATION
3111 020120 000402 BR 30 ;CONTINUE
3112 020122 016700 161232 20; MOV CLOCKP,R0 ;GET ADR OF CLOCK INFORMATION
3113
3114 020126 132767 000001 161215 30; BITB #BIT0,TFLAG ;SETUP FOR BKMOD ?
3115 020134 001403 BEQ 40 ;NO, CONTINUE
3116 020136 017701 161126 MOV #BKPTR,R1 ;YES, GET ADR OF THE BKMOD
3117 020142 000402 BR 50 ;CONTINUE
3118
3119 020144 016701 161344 40; MOV CSTART,R1 ;GET ADR OF MODULE DOING EOP
3120 020150 012702 026522 50; MOV #MODQ,R2 ;GET ADR OF MODULE ADDRESSES
3121 020154 012703 000004 MOV #4,R3 ;SET POINTER TO 1ST MODULE TIME ENTRY
3122
3123 020160 020122 60; CMP R1,(R2)+ ;HAS THE MODULE TIME BEEN FOUND?
3124 020162 001406 BEQ 70 ;YES, CONTINUE
3125 020164 062703 000004 ADD #4,R3 ;ADD 4 TO THE TABLE ADDRESS
3126 020170 022703 000360 CMP #240,,R3 ;AT THE END OF THE TABLE?
3127 020174 063171 BGT 60 ;NO, KEEP SEARCHING
3128 020176 000453 BR 90 ;SOMETHING'S WRONG--GET OUT
3129
3130 020200 060003 70; ADD R0,R3 ;GET ADR OF CLOCK INFO TABLE
3131 020202 011104 MOV (R3),R4 ;GET TIME OF MODULE'S LAST PASS
3132 020204 064767 177446 JSR PC,RUNTYM ;GET AND SET UP RUNTIME INFORMATION
3133 020210 016700 161200 MOV RTIME,R0 ;PUT THE RUNTIME IN REG. 0
3134 020214 160400 SUB R4,R0 ;GET THE MODULE'S ELAPSED TIME
3135 020216 105767 161127 TSTB TFLAG ;IS IT ERROR TIME ?
3136 020222 100415 BMI 00 ;YES, CONTINUE
3137 020224 016713 161164 MOV RTIME,(R3) ;NO, UPDATE THE MODULE'S PASS TIME
3138 020230 016763 161162 000002 MOV RTIMX,2(R3) ;AND THE EXTENDED BITS
3139
3140 020236 132767 000001 161105 BITB #BIT0,TFLAG ;SETUP FOR BKMOD ?
3141 020244 001030 BNE 00 ;YES, ALL DONE, RETURN
3142 020246 032777 010000 160722 BIT #BIT12,#SR ;NO, PRINT THE EOP DATA ?
3143 020254 001424 BEQ 00 ;NO, RETURN
3144 020256 010067 161142 00; MOV R0,PTIME ;PUT THE ELAPSED TIME INTO PTIME
    
```

```

3145 020262 005067 161130 CLR RTIMX ;THIS WILL ALWAYS BE 0
3146 020266 004767 177016 JSR PC,HMS ;CONVERT IT TO HOURS, MINUTES, AND SECONDS
3147 020272 001424 PTIME ;GIVEN THE TIME IN SECONDS
3148 020274 001416 RTIMX ;
3149 020276 116767 161106 005615 MOVB #MINITS,MMINS ;LOAD 10'S COLUMN OF MINUTES
3150 020304 116767 161101 005610 MOVB #MINITS+1,MMINS+1 ;LOAD 1'S COLUMN OF MINUTES
3151 020310 116767 161074 005604 MOVB #SEKNS,MSECS ;LOAD 10'S COLUMN OF SECONDS
3152 020320 116767 161067 005577 MOVB #SEKNS+1,MSECS+1 ;LOAD 1'S COLUMN OF SECONDS
3153
3154 020326 004767 174420 90; JSR PC,RST04 ;RESTORE REGS. 0-4
3155 020332 000207 RTS ;RETURN
    
```

```

3156 ; THIS ROUTINE WILL SHUT OFF THE CLOCK (IF AVAILABLE) TO KEEP
3157 ; IT FROM INTERRUPTING DURING THE MVCODE ROUTINE.
3158 ; CALL VIA: JSR PC,CLKOFF
3159
3160 020334 105767 161013 CLKOFF: TSTB CLOCK ;IS THERE A CLOCK ON THE SYSTEM ?
3161 020340 001425 BFC 48 ;NO, RETURN
3162 020342 132767 000001 161003 BITB #BIT0,CLOCK ;YES, THE KW11=L ?
3163 020350 001004 BNE 18 ;YES, CONTINUE
3164 020352 132767 000002 160773 BITB #BIT1,CLOCK ;NO, HOW ABOUT THE KW11=P ?
3165 020360 001003 BNE 26 ;YES, CONTINUE
3166
3167 020362 017700 160770 18: MOV @CLOCKL,R0 ;GET THE BEGIN ADDRESS OF CLOCK MODULE
3168 020366 000402 BR 38 ;CONTINUE
3169 020370 017700 160764 28: MOV @CLOCKP,R0 ;GET THE BEGIN ADDRESS OF CLOCK MODULE
3170
3171 020374 016067 000006 161016 38: MOV 6(R0),CLKADR ;SAVE THE CLOCK ADDRESS
3172 020402 017067 000006 161012 MOV @6(R0),CLKHLD ;SAVE CONTENTS OF CLOCK CSR
3173 020410 005070 000006 CLR @6(R0) ;KILL THE CLOCK
3174 020414 000207 RTS PC ;RETURN
3175
3176
3177
3178 ; THIS ROUTINE WILL TURN THE CLOCK BACK ON IF IT WAS TURNED
3179 ; OFF BY CLKOFF.
3180 ; CALL VIA: JSP PC,CLKON
3181
3182 020416 105767 160731 CLKON: TSTB CLOCK ;IS THERE A CLOCK ON THE SYSTEM ?
3183 020422 001403 BFC 18 ;NO, RETURN
3184 020424 016777 160772 160766 MOV CLKHLD,@CLKADR ;RESTORE THE CLOCK STATUS
3185 020432 000207 RTS PC ;RETURN
3186
3187
3188 ; THIS IS THE CALLING SEQUENCE FOR THE CKHUNG ROUTINE
3189
3190 020434 016706 160512 HUNGCK: MOV SPSAV,SP ;RESTORE MONITOR STACK POINTER
3191 020440 005067 157332 CLR PSW ;PROCESSOR TO STATUS 0
3192 020444 005067 161044 CLR CSTART ;INDICATE NO MODULES DOING EOP
3193 020450 004767 000006 JSR PC,CKHUNG ;GO CHECK FOR HUNG MODULES
3194 020454 000240 NOP
3195 020456 000167 164476 JMP ENDSVG ;GO CHECK FOR OTHER MODULES RUNNING

```

```

3196 ;*****
3197 ;THIS ROUTINE IS CALLED FROM THE "DSEND" OR HUNGCK ROUTINE AND CHECKS
3198 ;THAT ALL RUNNING MODULES HAVE COMPLETED AT LEAST ONE PASS DURING A
3199 ;SPECIFIED TIME INTERVAL. IF IT DETECTS A "HUNG" MODULE, THE
3200 ;MODULE IS DROPPED AND AN ERROR MESSAGE REPORTED.
3201
3202 020462 004467 174252 CKHUNG: JSR R4,SAV04 ;GO SAVE R0 = R4
3203 020466 132767 000001 160657 BITB #BIT0,CLOCK ;USING THE KW11=L CLOCK ?
3204 020474 001007 BNE 18 ;BR IF YES
3205 020476 132767 000002 160647 BITB #BIT1,CLOCK ;USING THE KW11=P CLOCK ?
3206 020504 001513 BEQ 78 ;BR IF NOT - SOMETHING'S SCREWED ??
3207 020506 016702 160646 MOV CLOCKP,R2 ;GET POINTER TO TIMING INFO
3208 020512 000402 BR 28 ;CONTINUE ON
3209 020514 016702 160636 18: MOV CLOCKL,R2 ;GET POINTER TO TIMING INFO
3210 020520 016203 MOV R2,R3 ;COPY R2
3211 020522 062703 000004 ADD #4,R3 ;R3 POINTS TO TABLE OF PASS TIMES
3212 ;IN THE "TIMER" MODULE
3213 020526 012700 026522 MOV #MOD0,R0 ;GET POINTER TO MODULE ADDR TABLE
3214 020532 005067 160670 CLR MAXTIM ;ZERO LARGEST ELAPSED TIME
3215 020536 012001 38: MOV (R0)+,R1 ;PUT MODULE BEGIN ADDR IN R1
3216 020540 001442 BFC 48 ;BR IF DONE ALL MODULES
3217 020542 026701 160746 CMP CSTART,R1 ;IS IT THE MODULE DOING THE EOP ?
3218 020546 001467 BEQ 68 ;YES, SKIP IT
3219 020550 022761 144000 000020 CMP #144000,STAT(R1) ;IS IT AN IOMODR ?
3220 020556 001463 BEQ 66 ;BR IF YES
3221 020560 032761 040000 000020 BIT #BIT14,STAT(R1) ;IS IT SELECTED ??
3222 020566 001457 BEQ 66 ;BR IF NOT
3223 020570 032761 020000 000020 BIT #BIT13,STAT(R1) ;WAS IT ALREADY DROPPED ??
3224 020576 001053 BNE 68 ;BR IF YES
3225 020600 005761 000020 TST STAT(R1) ;IS IT A RMOD ?
3226 020604 100050 BPL 68 ;YES, FORGET IT
3227 020606 022761 142000 000020 CMP #142000,STAT(R1) ;IS IT AN IOMODP ?
3228 020614 001444 BEQ 68 ;YES, FORGET IT
3229 020616 016204 000002 MOV 2(R2),R4 ;GET TIME FROM THE TIMER
3230 020622 161304 SUB (R3),R4 ;CALCULATE TIME SINCE LAST ENDPAS
3231 020624 062703 000004 ADD #4,R3 ;POINT TO THE NEXT TIME ENTRY
3232 020630 020467 160572 CMP R4,MAXTIM ;IS THIS ELAPSED TIME BIGGER ?
3233 020634 010740 BLOS 38 ;NO, CONTINUE
3234 020636 010105 MOV R1,R5 ;YES, SAVE THE MODULE POINTER
3235 020640 010467 160562 MOV R4,MAXTIM ;UPDATE THE MAXTIM
3236 020644 000734 BR 38 ;CONTINUE
3237 020646 026767 160510 160552 48: CMP HNGTIM,MAXTIM ;IS THE MODULE HUNG ??
3238 020654 100027 BPL 78 ;IF NOT, RETURN
3239 020656 010501 MOV R5,R1 ;YES, INDICATE MODULE TO BE DROPPED
3240 020660 004767 000070 PC,DRPHNG ;GO DROP IT AND REPORT ERROR
3241 020664 105767 160134 TSTB STOP ;WAITING TO RELOCATE ?
3242 020670 001421 BEQ 78 ;NO, RETURN
3243 020672 012700 026522 MOV #MOD0,R0 ;SET POINTER TO MODULE ADDR TABLE
3244 020676 016204 000002 MOV 2(R2),R4 ;YES, GET THE PRESENT TIME
3245 020702 016201 177776 MOV -2(R2),R1 ;AND TIME EXTENDED BITS
3246 020706 010203 MOV R2,R3 ;GET POINTER TO TIME INFO
3247 020710 062703 000004 ADD #4,R3 ;POINT TO PASTIME INFO
3248
3249 020714 005720 58: TST (R0)+ ;ALL DONE ?
3250 020716 001406 BEQ 78 ;YES, RETURN
3251 020720 010423 MOV R4,(R3)+ ;UPDATE ALL MODULE PASTIMES

```

```

3252 020722 010123          MOV    R1,(R3)+      ;AND TIME EXTENDED RITS
3253 020724 000773          BR     5$           ;CONTINUE
3254 020726 062703 000004   6$:   ADD    #4,P3      ;POINT TO NEXT PASS TIME ENTRY
3255 020732 000701          BR     3$           ;GO DO IT AGAIN
3256 020734 004767 174012   7$:   JSR    PC,RST04    ;GO RESTORE R0 = R4
3257 020740 105767 160051   TSTB  MODCNT       ;ANY MODULES STILL RUNNING ??
3258 020744 001002          BNE   8$           ;BR IF YES
3259 020746 000167 166076   8$:   JMP    CTRLCB      ;GO TERMINATE THIS "RUN"
3260 020752 000207          RTS    PC          ;RETURN TO CALLING ROUTINE
    
```

```

3261          ;*****
3262          ;THIS ROUTINE IS CALLED FROM THE "CKHUNG" ROUTINE AND IT IS USED
3263          ;TO DROP THE HUNG MODULE AND REPORT THE ERROR ON THE CONSOLE
3264          ;TTY. IF THE "TIMER" MODULE IS FOUND TO BE HUNG THEN THE ROUTINE
3265          ;DISABLES ANY FURTHER TIMING CHECKS USING THE CLOCK
3266          ;BY CLEARING THE BYTE TAGGED "CLOCK"
3267
3268 020754 052761 020000 000020 DRPHNG: RIS    #RIT13,STAT(R1) ;SET THE DROPPED BIT
3269 020762 032761 010000 000020          BIT    #RIT12,STAT(R1) ;TO *ODX MODULE ??
3270 020770 001402          BEQ    1$           ;BR IF NOT
3271 020772 105367 160024          DECB  MDXCNT       ;DECREMENT COUNT OF IOMODX MODULES
3272 020776 105367 160013   1$:   DECB  MODCNT       ;DECREMENT COUNT OF PUNNING MODULES
3273 021002 021201          CMP    (R2),R1     ;DID WE STOP THE TIMER MODULE ??
3274 021004 001014          BNE   2$           ;BR IF NOT
3275 021006 105067 160341          CLRB  CLOCK       ;DISABLE ALL TIMING CHECKS
3276 021012 010046          MOV    R0,-(SP)    ;TYPEN SCREWS YOU IF YOU DON'T SAVE R0
3277 021014 004567 166330          JSR    R5,TYPEN   ;GO REPORT DROPPING THE TIMER MODULE
3278          CRCR
3279 021022 022744          CRCR
3280 021024 024230          TIMX1
3281 021026 022744          CRCR
3282          0
3283 021032 012600          MOV    (SP)+,R0    ;RESTORE R0
3284 021034 000207          RTS    PC          ;RETURN TO "CKHUNG"
3285
3286 021036 021267 160226   2$:   CMP    (R2),BKPTR   ;DID WE DRDP AN ACTIVE BKMOD ?
3287 021042 001002          BNE   3$           ;BR IF NOT
3288 021044 105067 157743          CLRB  BRAKE       ;CLEAR THE BRAKE
3289 021050 004567 172560   3$:   JSR    R5,FILLNM   ;GO GET MOD NAME SET UP
3290 021054 026026          TIMX+1          ;POINTER TO ERROR MESSAGE
3291 021056 004767 176770          JSR    PC,ERRIYM   ;GO SETUP TIME INFORMATION
3292 021062 010046          MOV    R0,-(SP)    ;TYPEN SCREWS YOU IF YOU DON'T SAVE R0
3293 021064 004567 166260          JSR    R5,TYPEN   ;REPORT THAT WE DROPPED IT
3294          CRCR
3295 021072 022744          CRCR
3296 021074 026025          TIMX
3297 021076 026066          TOTIM
3298 021100 022744          CRCR
3299 021102 000000          0
3300 021104 012600          MOV    (SP)+,R0    ;RESTORE R0
3301 021106 000207          RTS    PC          ;RETURN TO "CKHUNG" ROUTINE
3302
    
```

```

3303 021110 016767 160056 160050 LPON: MOV LPS,TPS
3304 021116 016767 160052 160044 MOV LPB,TPB
3305 021124 012737 001526 000200 MOV #TTISR,##200
3306 021132 012737 000200 000202 MOV #PRTY4,##202
3307 021140 000167 167512 JMP COMCO3
3308 021144 177564 LPWK1: 177564
3309 021146 177566 LPWK2: 177566
;GET WRITE BUFFER ADDRESS SERVICE
3310
3311
3312 021150 010146 GWBUF,; MOV R1,=(SP) ;SAVE REGISTER
3313 021152 010246 MOV R2,=(SP)
3314 021154 017601 000004 MOV #4(SP),R1 ;GET ADDRESS OF MODULE
3315 021160 010102 MOV R1,R2
3316 021162 062701 000074 ADD #WBUFFA,R1
3317 021166 016721 160040 MOV WBUFF,(R1)+ ;LOAD ADDRESS OF WRITE BUFFER
3318 021172 016721 160036 MOV WBFEA,(R1)+ ;LOAD EXTENDED ADDRESS BITS
3319 021176 012111 MOV (R1)+,(R1) ;MOVE SIZE REQUESTED INTO SIZE ALLOWED
3320 021200 021167 160032 CMP (R1),MXWRF ;REQUESTED SIZE OK?
3321 021204 101402 BLOS 16 ;YES= BRANCH
3322 021206 016711 160044 MOV MXWBF,(R1) ;NO= SET ALLOWED SIZE
3323 021212 105767 157603 16: TSTB MDXCTR ;IF NOT YET SETUP, SKIP
3324 021216 100410 BMI 26
3325 021220 126267 000005 157575 CMPA XFLAG(2),WBFLG ;IF SAME, HAS ALREADY USED THIS BANK
3326 021226 001404 REQ 26
3327 021230 105162 000005 CDME XFLAG(2) ;IF NOT, SET FLAG TO SAME
3328 021234 105367 157561 DECB MDXCTR ;AND COUNT DOWN MODULES
3329 021240 012602 MOV (SP)+,R2 ;RESTORE REGISTERS
3330 021242 012601 MOV (SP)+,R1
3331 021244 062716 000002 ADD #2,(SP) ;SETUP RETURN PC
3332 021250 105367 157561 DECB ROT
3333 021254 100005 BPL 36
3334 021256 116767 160055 157551 MOVR ROTN,ROT ;GO CHANGE WRITE BUFFER ADDRESS
3335 021264 004767 175014 JSP PC,ROTBUF ;RETURN
3336 021270 000002 36: RTI
3337
3338
    
```

```

3339 ;ROUTINE TO GET 18 BIT PHYSICAL ADDRESS OF VIRTUAL ADDRESS GIVEN,
3340 ;TOP 2 BITS ARE ALLIGNED TO POSITIONS 4 AND 5
3341 ;ASSUMES KERNEL I-SPACE ADDRESS IN CURRENT MAP
3342
3343 021272 010146 GETPA,; MOV R1,=(SP) ;SAVE REGISTERS
3344 021274 010246 MOV R2,=(SP)
3345 021276 017601 000004 MOV #4(SP),R1 ;GET TABLE ADDRESS
3346 021302 012111 MOV (R1)+,(R1) ;VA INTO PA LOC IN TABLE
3347 021304 105767 157774 TSTB KTPRES ;KT11 IN USE?
3348 021310 001003 BNE 16 ;YES= BRANCH
3349 021312 005061 000002 CLR 2(R1) ;NO= CLEAR EA BITS
3350 021316 000436 BR 36 ;AND EXIT
3351
3352 021320 000311 16: SWAB (R1)
3353 021322 006211 ASR (R1)
3354 021324 006211 ASP (R1)
3355 021326 006211 ASR (R1)
3356 021330 006211 ASR (R1)
3357 021332 042711 177761 RIC #177761,(R1) ;PAGE BITS ALLIGNED
3358 021336 062711 172340 ADD #KIPAR,(R1) ;KIPAR ADDRESS FORMED
3359 021342 017111 000000 MOV #0(R1),(R1) ;GET CONTENTS
3360 021346 017702 000006 MOV #6,R2 ;SHIFT 6 LEFT
3361 021352 006311 26: ASL (R1)
3362 021354 006161 000002 ROL 2(R1)
3363 021360 005302 DEC R2
3364 021362 001373 BNE 26
3365 021364 014102 MOV =(R1),R2 ;GET VA
3366 021366 042702 160000 RIC #160000,R2 ;CLEAR PAGE BITS
3367 021372 005721 TST (R1)+
3368 021374 060221 ADD R2,(R1)+ ;CREATE 16 BIT PA
3369 021376 005511 ADC (R1)
3370 021400 042711 177774 BIC #177774,(R1) ;ALIGN BITS IN POSITIONS 4+5
3371 021404 006311 ASL (R1)
3372 021406 006311 ASL (R1)
3373 021410 006311 ASL (R1)
3374 021412 006311 ASL (R1)
3375 021414 012602 36: MOV (SP)+,R2 ;RESTORE REGISTERS
3376 021416 012601 MOV (SP)+,R1
3377 021420 062716 000002 ADD #2,(SP)
3378 021424 000002 RTI ;RETURN
3379
3380
    
```

```

3381                                     ;THIS ROUTINE ALLOWS RH70 DEVICES TO BE RUN WITH THE STANDARD MONITOR.
3382
3383
3384 021426 010146          MAP22: MOV      R1, -(SP)          ;SAVE REG. 1
3385 021430 017601          MOV      @2(SP),R1          ;GET THE 1ST ADDRESS OF THE TABLE
3386 021434 012161          MOV      (R1)+,2(R1)        ;GIVE BACK THE SAME 18 BITS
3387 021440 011161          MOV      (R1),4(R1)        ;THAT WERE GIVEN TO BE CONVERTED
3388 021444 006261          ASR      4(R1)          ;SHIFT BACK FROM POSITIONS 4 AND 5
3389 021450 006261          ASR      4(R1)          ;TO POSITIONS 0 AND 1
3390 021454 006261          ASR      4(R1)          ;
3391 021460 006261          ASR      4(R1)          ;
3392 021464 012601          MOV      (SP)+,R1          ;RESTORE REG. 1
3393 021466 062716          ADD      #2,(SP)          ;SKIP OVER VA TO GET THE RETURN PC
3394 021472 000002          RTI              ;RETURN
    
```

```

3395                                     ;MOVE CODE TO NEW OFFSET, AFTER CHECKING FOR CODE ALREADY IN
3396                                     ;MIDST OF BEING MOVED. MAP KT11 TO NEW CODE AND RETURN TO IT.
3397                                     ;ALL VALUES USED EXCEPT OFFSTD ARE ASSUMED CORRECT IN BANK 0 COPY
3398                                     ;*** STACK POINTER IS REPOSITIONED TO SP00T BEFORE RETURNING
3399                                     ;MVCODS ENTRY POINT IS USED TO LEAVE OFFSET AS PREVIOUSLY SETUP-
3400                                     ;NOTE, HOWEVER, THAT MAPRET MUST BE LOADED WITH THE RETURN ADDRESS
3401                                     ;BEFORE CALLING. MVCODS SHOULD NOT BE CALLED UNLESS KTPRES IS SET
3402                                     ;NOTE THAT MVCODE DISABLES THE TTY KEYBOARD
3403
3404 021474 105767 157604    MVCODE: TSTB     KTPRES          ;KT11 IN USE?
3405 021500 001001          BNE      ++4          ;YES- BRANCH
3406 021502 000207          RTS      PC          ;NO- RETURN (MOVING NOT ALLOWED W/O KT11)
3407 021504 016700 157712    MOV      CLKHLD,R0        ;SAVE CONTENTS OF CLOCK CSR
3408 021510 016705 157704    MOV      CLKADR,R5        ;SAVE ADR OF CLOCK CSR
3409 021514 016703 157524    MOV      OFFSTD,R3        ;SAVE DESTINATION OFFSET
3410 021520 011604          MOV      (SP),R4         ;SAVE RETURN PC
3411 021522 005037 177572    CLR      ##SSR0          ;TURN OFF KT11
3412 021526 004767 170416    JSP      PC,KTSET0        ;MAP DOWN TO BANK 0
3413 021532 052737 000001    BIS      #BIT0,##SSR0     ;TURN ON KT11
3414 021540 010067 157656    MOV      R0,CLKHLD        ;STORE CONTENTS OF CLOCK CSR IN BANK 0
3415 021544 010567 157650    MOV      R5,CLKADR        ;STORE ADR OF CSR IN BANK 0
3416 021550 010367 157470    MOV      R3,OFFSTD        ;SET UP DESTINATION OFFSET IN BANK 0
3417
3418 021554 010467 157472    MOV      R4,MAPRET        ;SETUP RETURN PC
3419 021560 105067 157561    CLRFB   MOVPLG          ;CLEAR FLAG FOR 'ROTATED' MESSAGE
3420 021564 005077 157372    MVCODS: CLRFB   @TKS          ;SHUT OUT TTY
3421 021570 004767 173172    JSR      PC,CLRQUS
3422 021574 105067 157230    MVCOD1: CLRFB   XFRPLG
3423 021600 012737 077406    MOV      #77406,##KIPDR2 ;
3424 021606 012737 077406    MOV      #77406,##KIPDR2 ;
3425 021614 105767 157505    MOVING  TSTB          ;CODE PARTIALLY RELOCATED?
3426 021620 001406          BEQ      18            ;NO- BRANCH
3427 021622 100067          BPL      48            ;IF PARTIALLY RELOCATED, BRANCH TO
3428                                     ;FINISH RELOCATION
3429 021624 016767 157416 157410    MOV      OFFSTM, OFFSET  ;IF FLAG NEGATIVE, INDICATES MOVE IS
3430 021632 105067 157467          CLRFB   MOVING        ;DONE BUT POINTERS NOT YET RESTORED
3431 021636 026767 157402 157376 18:    CMP      OFFSTD,OFFSET
3432 021644 001004          BNE      18
3433 021646 105167 157473          COMB    MOVPLG          ;SET FLAG NOT TO PRINT MESSAGE
3434 021652 000167 000534          JMP      MVMAP
3435 021656 016767 157362 157362 118:    MOV      OFFSTD, OFFSTM  ;PUT DESTINATION OFFSET INTO SAVE LOCATION
3436 021664 026767 157356 157350    CMP      OFFSTM, OFFSET  ;TO PREVENT LOSING VALUE IN MIDST OF MOVING
3437 021672 101012          BHI      28            ;IF MOVING CODE UP, BRANCH
3438 021674 016737 157342 172344    MOV      OFFSET,##KIPAR2 ;IF MOVING DOWN, SETUP TO
3439 021702 016737 157340 172346    MOV      OFFSTM,##KIPAR3 ;DO LOWEST BANK FIRST
3440 021710 112767 000001 157412    MOVFB   #1,MVDOWN
3441 021716 000450          BR      MVCOD2
3442 021720 016737 157300 172344 28:    MOV      LCCR12,##KIPAR2 ;IF MOVING UP, DO HIGHEST BANK FIRST
3443 021726 032737 000177 172344    BIT      #177,##KIPAR2
3444 021734 001003          BNE      38
3445 021736 162737 000200 172344    SUB     #200,##KIPAR2
3446 021744 042737 000175 172344 38:    BIC     #177,##KIPAR2  ;IF 4K BOUNDARY, DROP TO START OF LAST BANK OF CODE
3447 021752 011737 172344 172346    MOV     ##KIPAR2,##KIPAR3
3448 021760 066737 157256 172344    ADD     OFFSET, ##KIPAR2
3449 021766 066737 157254 172346    ADD     OFFSTM,##KIPAR3
3450 021774 105067 157330          CLRFB   MVDOWN
    
```

```

3451 022000 000417          BR      MVCOD2      ;GO TO ROUTINE THAT ACTUALLY MOVES THE CODE
3452 022002 112767 000001 157020 48i    MOV      #1,XPRFLG      ;INDICATE ANOTHER PAGE MUST BE MADE
3453 022010 016737 157234 172344      MOV      MOVBNK,##KIPAR2 ;TO DO DESIRED MOVE AFTER CLEANING
3454 022016 016737 157226 172346      MOV      MOVBNK,##KIPAR3 ;UP LAST ONE
3455 022024 156737 157212 172346      SUB      OFFSET,##KIPAR3
3456 022032 066737 157210 172346      ADD      OFFSTM,##KIPAR3
3457 022040 105067 157245      MVCOD2i CLR      RETRY
3458 022044 013767 172344 157176      MOV      ##KIPAR2,MOVBNK ;SAVE CURRENT BANK BEING MOVED
3459 022052 112767 000001 157245      MOV      #1,MOVING      ;INDICATE MOVE IN PROCESS
3460 022060 012701 040000      MVRTRYi MOV      #40000,R1
3461 022064 012702 060000      MOV      #60000,R2
3462 022070 005737 172346      TST     ##KIPAR3
3463 022074 001013          BNE     28
3464 022076 012122          MOV      (R1)+,(R2)+ ;COPY VECTOR AREA DOWN TO BANK 0
3465 022100 032702 001000      BIT     #1000,R2
3466 022104 001774          BEQ     18
3467 022106 013737 041270 061270      MOV      ##BKPTR+40000,##BKPTR+60000 ;COPY BKPTR TO 0
3468 022114 012701 055000      MOV      #55000,R1
3469 022120 012702 075000      MOV      #75000,R2
3470 022124 012122          MOV      (R1)+,(R2)+ ;MOVE CODE
3471 022126 032701 017777      BIT     #17777,R1
3472 022132 001374          RNE     26
3473 022134 012701 040000      MOV      #40000,R1
3474 022140 012702 060000      MOV      #60000,R2
3475 022144 005737 172346      TST     ##KIPAR3
3476 022150 001014          BNE     58
3477 022152 021112          CMP     (R1),(R2)
3478 022154 001402          BEQ     48
3479 022156 000167 000402      JMP     MVERR
3480 022162 022122          CMP     (R1)+,(R2)+
3481 022164 032702 001000      BIT     #1000,R2
3482 022170 001770          BEQ     38
3483 022172 012701 055000      MOV      #55000,R1
3484 022176 012702 075000      MOV      #75000,R2
3485 022202 021112          CMP     (R1),(R2) ;NEW COPY OK?
3486 022204 001402          BEQ     68 ;YES, CONTINUE
3487 022206 000167 000352      JMP     MVERR ;NO- BRANCH TO REPORT ERROR
3488 022212 022122          CMP     (R1)+,(R2)+
3489 022214 032701 017777      BIT     #17777,R1
3490 022220 001370          BNE     58
3491 022222 026767 157020 157012      CMP     OFFSTM,OFFSET
3492 022230 101413          BLOS   78
3493 022232 162737 000200 172344      SUB     #200,##KIPAR2
3494 022240 162737 000200 172346      SUB     #200,##KIPAR3
3495 022246 023767 172346 156772      CMP     ##KIPAR3,OFFSTM
3496 022254 103271          BHS    MVCOD2
3497 022256 000415          BR     MVCOD3
3498 022260 062737 000200 172344      ADD     #200,##KIPAR2
3499 022266 062737 000200 172346      ADD     #200,##KIPAR3
3500 022274 016701 156746      MOV     OFFSTM,R1
3501 022300 066701 156720      ADD     LCOR12,R1
3502 022304 023701 172346      CMP     ##KIPAR3,R1
3503 022310 103653          BLO    MVCOD2
3504 022312 112767 177777 157005      MVRB   #-1,MOVING ;INDICATE MOVE DONE BUT POINTERS
3505 022320 016767 156722 156714      MOV     OFFSTM,OFFSET ;NOT YET RESTORED
3506 022326 016737 156710 172344      MOV     OFFSET,##KIPAR2
    
```

```

3507 022334 012701 001242          MOV     #OFFSET,R1
3508 022340 062701 040000          ADD     #40000,R1
3509 022344 016711 156672          MOV     OFFSET,(R1) ;SET OFFSET VALUE INTO NEW COPY
3510 022350 105067 156751          CLR    MOVING
3511 022354 105767 156750          TST    MVDOWN
3512 022360 001402          BFO    MVMMSG
3513 022362 004767 002304          JSR    PC,LOADWC ;RELOAD WORST CASE FROM LCOR12+OFFSET
3514                                     ;TO HCOR12
3515 022366 005067 156466          MVMMSGi CLR    IOBKID
3516 022372 112767 000001 156413      MOV    #1,BRAKE
3517 022400 105767 156424          TST    XPRFLG ;FINAL MOVE NOT YET DONE?
3518 022404 001402          REQ    MVMAP
3519 022406 000167 177162          JMP    MVCOD1 ;YES- DO DESIRED MOVE NOW THAT PREVIOUS ONE
3520                                     ;HAS BEEN CLEANED UP
3521 022412 016704 156634          MVMAPi MOV    MABRET,R4 ;GET RETURN PC
3522 022416 016701 156620          MOV    OFFSET,R1 ;GET NEW OFFSET
3523 022422 062701 000200          ADD     #200,R1
3524 022426 012702 172302          MOV     #KIPDR1,R2
3525 022432 012703 000006          MOV     #6,R3
3526 022436 012712 077406          18i    MOV     #77406,(R2) ;SET PDR TO 4K,RW,UP
3527 022442 010162 000040          MOV     R1,40(R2) ;MAP PAR TO DESIRED 4K
3528 022446 062702 000002          ADD     #2,R2 ;MOVE POINTER TO NEXT PAGE
3529 022452 062701 000200          ADD     #200,R1 ;CHANGE TO NEXT 4K VALUE
3530 022456 005303          DEC     R3
3531 022460 001366          BNE     18
3532 022462 116701 156657          MOV    MOVFLG,R1 ;SAVE THE FLAG
3533 022466 016737 156550 172340      MOV    OFFSET,##KIPAR0 ;CHANGE TO NEW CODE
3534 022474 012706 027212          MOV     #SPROT,SP
3535 022500 005037 177776          CLR    #4PSW
3536 022504 105701          TST    R1 ;OK TO PRINT MESSAGE ?
3537 022506 100425          BMI    28 ;NO, CONTINUE
3538 022510 032777 001000 156460      BIT     #BIT9,##R ;YES, BUT IS IT INHIBITED ?
3539 022516 001021          BNE     28 ;YES, CONTINUE
3540 022520 104406 024446          MSGS,RELMSG ;CODE RELOCATED TO*
3541 022524 004567 172142          JSR    R5,TO18S ;GET 18 BIT ADDRESS
3542 022530 001242          OFFSET
3543 022532 001206          MEMPA
3544 022534 001210          MEMEA
3545 022536 016746 156444          MOV    MEMPA,=(SP)
3546 022542 016746 156442          MOV    MEMEA,=(SP)
3547 022546 004767 170564          JSR    PC,ITDAX ;TYPE ADDRESS
3548 022552 012746 022772          MOV    #ACRLF,=(SP)
3549 022556 004767 164604          JSR    PC,TYPE
3550 022562 010407          28i    MOV    R4,PC ;RETURN FROM MYCODE CALL
3551
3552
3553 022564 105067 156270          MVERRi CLR    IOBKID
3554 022570 112767 000001 156215      MOV    #1,BRAKE
3555 022576 010267 156402          MOV    R2,MEMVA
3556 022602 104406 024420          MSGS,RELERR ;*RELOCATION ERROR AT*
3557 022606 104413 001204          GETPAS,MEMVA
3558 022612 016746 156370          MOV    MEMPA,=(SP)
3559 022616 016746 156366          MOV    MEMEA,=(SP)
3560 022622 004767 170510          JSR    PC,ITDAX ;TYPE ADDRESS
3561 022626 012746 022772          MOV    #ACRLF,=(SP)
3562 022632 004767 164530          JSR    PC,TYPE
    
```

3563	022636	105267	156467			INCB	RETRY		
3564	022642	126727	156463	000003		CMPB	RETRY,#3		
3565	022650	002002				BGE	68		
3566	022652	000167	177202			JMP	MVRTRY		
3567	022656	104406	024100		68:	MSG8,FATAL			,"UNABLE TO RELOCATE- FATAL, RELOAD,"
3568	022662	104406	023746			MSG8,HLT			,"HALT"
3569	022666	000000			78:	HALT			
3570	022670	000776				BR	78		

3571 ;PURE MESSAGES  
3572  
3573 022672 042045 041505 054057 TITLE: .ASCIZ '%DEC/X11. DXQAB=M STANDARD MONITOR'  
3574 022700 030461 020056 042040  
3575 022706 050530 041101 046455  
3576 022714 020040 052123 047101  
3577 022722 040504 042122 046440  
3578 022730 047117 052111 051117  
3579 022736 000  
3580 022737 045 000056 DOT: .ASCIZ '%.'  
3581 022742 000045 CR: .ASCIZ '%'  
3582 022744 022445 000 CRCR: .ASCIZ '%%'  
3583 022747 040 052101 000 AAT: .ASCIZ '% AT'  
3584 022753 045 052522 020116 SUMMARY: .ASCIZ '%RUN SUMMARY'  
3585 022760 052523 046515 051101  
3586 022766 000131  
3587 022772  
3588 022770 041536 000045 ACRLF=,+2  
3589 022774 022445 050440 042525 CTRLC: .ASCIZ '%C%'  
3590 023002 042525 047440 042526 QOVFL: .ASCIZ '%% QUEUE OVERFLOW --- SYSTEM CRASH%'  
3591 023010 043122 047514 020127  
3592 023016 026455 020055 054523  
3593 023024 052123 046505 041440  
3594 023032 040522 044123 000045  
3595 023040 022445 044440 053116 INVTRP: .ASCIZ '%% INVALID TRAP CALL --- SYSTEM CRASH%'  
3596 023046 046101 042111 052040  
3597 023054 040522 020120 040503  
3598 023062 046114 026440 026455  
3599 023070 051440 051531 042524  
3600 023076 020115 051103 051501  
3601 023104 022510 000  
3602 023107 045 047111 040526 INVCMD: .ASCIZ '%INVALID COMMAND%'  
3603 023114 044514 020104 047503  
3604 023122 046515 047101 022504  
3605 023130 000  
3606 023131 045 052522 020116 INVCDD1: .ASCIZ '%RUN MODE --- ONLY A CNL C IS PERMITTED%'  
3607 023136 047515 042504 026440  
3608 023144 026455 047440 046116  
3609 023152 020131 020101 047103  
3610 023160 046124 041440 044440  
3611 023166 020123 042520 046522  
3612 023174 052111 042524 022504  
3613 023202 000  
3614 023203 045 051120 043517 INVCDD2: .ASCIZ '%PROGRAM CANNOT GO THAT HIGH%'  
3615 023210 040522 020115 040503  
3616 023216 047116 052117 043440  
3617 023224 020117 044124 052101  
3618 023232 044040 043511 022510  
3619 023240 000  
3620 023241 045 047516 053440 INVCDD3: .ASCIZ '%NO WAY! MEMORY MANAGEMENT MUST BE IN USE%'  
3621 023246 054501 020041 020040  
3622 023254 042515 047515 054522  
3623 023262 046440 047101 043501  
3624 023270 046505 047105 020124  
3625 023276 052515 052123 041040  
3626 023304 020105 047111 052440

3627	023312	042523	000045						
3628	023316	047045	020117	047515	INVCDA:	.ASCIZ	'%NO MODULES SELECTED%'		
3629	023324	052504	042514	020123					
3630	023332	042523	042514	052103					
3631	023340	042105	000045						
3632	023344	044445	053116	046101	INVADRI	.ASCIZ	'%INVALID ADDR/DATA%'		
3633	023352	042111	040440	042104					
3634	023360	027522	040504	040524					
3635	023366	000045							
3636	023370	046445	042117	046125	INVNAM:	.ASCIZ	'%MODULE NAME NOT FOUND%'		
3637	023376	020105	040516	042515					
3638	023404	047040	052117	043040					
3639	023412	052517	042116	000045					
3640	023420	020045	020040	045050	YSER1:	.ASCIZ	'% (JAM,SRV,PBA,CHA,FLG/INT,WHAMI,CDATA,CTAG/CPU)A '		
3641	023426	046501	051454	053122					
3642	023434	050054	040502	041454					
3643	023442	040525	043054	043514					
3644	023450	044457	052116	053454					
3645	023456	040510	044515	041454					
3646	023464	040504	040524	041454					
3647	023472	040524	027507	050103					
3648	023500	024525	020045	000					
3649	023505	045	041113	043125	KBOFLO:	.ASCIZ	'%KBUF OFLO%'		
3650	023512	047440	046106	000117					
3651	023520	047445	047524	042440	CLR40:	.ASCIZ	'%TO EXERCISE LOAD MEDIUM YOU MUST CLEAR LOC 40%'		
3652	023526	042530	041522	051511					
3653	023534	020105	047514	042101					
3654	023542	046440	042105	052511					
3655	023550	020115	047531	020125					
3656	023556	052515	052123	041440					
3657	023564	042514	051101	046040					
3658	023572	041517	032040	022460					
3659	023600	000							
3660	023601	045	054523	020123	SYSERR:	.ASCIZ	'%SYS ERR (VIRT,PC,PSW,SP,VECT):'		
3661	023606	051105	020122	053050					
3662	023614	051111	027124	041520					
3663	023622	050054	053523	051454					
3664	023630	026120	042526	052103					
3665	023636	035051	020040	000					
3666	023643	045	040503	044103	CONN:	.ASCIZ	'%CACHE ENABLED%'		
3667	023650	020105	047105	041101					
3668	023656	042514	000104						
3669	023662	041445	041501	042510	COFFP:	.ASCIZ	'%CACHE DISABLED%'		
3670	023670	042040	051511	041101					
3671	023676	042514	000104						
3672	023702	022445	047520	042527	PWRFAI:	.ASCIZ	'%POWER FAILURE%'		
3673	023710	020122	040506	046111					
3674	023716	051125	022505	000045					
3675	023724	051045	047125	046440	MEMCHG:	.ASCIZ	'%RUN MEMORY TESTS%'		
3676	023732	046505	051117	020131					
3677	023740	042524	052123	000123					
3678	023746	044045	046101	022524	HLT:	.ASCIZ	'%HALT%'		
3679	023754	000045							
3680	023756	050045	051101	052111	PENABLI:	.ASCIZ	'%PARITY ENABLED%'		
3681	023764	020131	047105	041101					
3682	023772	042514	000104						

3683	023776	047045	020117	052113	NOKTI:	.ASCIZ	'%NO KTI%'		
3684	024004	030461	000						
3685	024007	045	052113	030461	KTDIS:	.ASCIZ	'%KTI OFF%'		
3686	024014	047440	043106	000					
3687	024021	045	047516	050040	NOPAR:	.ASCIZ	'%NO PARITY%'		
3688	024026	051101	052111	000131					
3689	024034	050045	051101	052111	PAOFF:	.ASCIZ	'%PARITY OFF%'		
3690	024042	020131	043117	000106					
3691	024050	022445	052113	030461	KTFATL:	.ASCIZ	'%KTI ABORT --- FATAL%'		
3692	024056	040440	047502	052122					
3693	024064	026440	026455	043040					
3694	024072	052101	046101	000045					
3695	024100	052445	040516	046102	FATAL:	.ASCIZ	'%UNABLE TO RELOCATE- FATAL, RELOAD %'		
3696	024106	020105	047524	051040					
3697	024114	046105	041517	052101					
3698	024122	026505	043040	052101					
3699	024130	046101	020056	042522					
3700	024136	047514	042101	022440					
3701	024144	000							
3702	024145	045	022445	025045	SYSBAD:	.ASCIZ	'%***** TOO MANY SYS ERRORS --- EXERCISE ABORTED%'		
3703	024152	025052	052040	047517					
3704	024160	046440	047101	020131					
3705	024166	054523	020123	051105					
3706	024174	047522	051522	026440					
3707	024202	026455	042440	042530					
3708	024210	041522	051511	020105					
3709	024216	041101	051117	042524					
3710	024224	022504	000045						
3711	024230	041445	047514	045503	TINXI:	.ASCIZ	'%CLOCK MODULE DROPPED. ALL TIMING FUNCTIONS DISABLED.%'		
3712	024236	046440	042117	046125					
3713	024244	020105	051104	050117					
3714	024252	042520	027104	040440					
3715	024260	046114	052040	046511					
3716	024266	047111	020107	052506					
3717	024274	041516	044524	047117					
3718	024302	020123	044504	040523					
3719	024310	046102	042105	022456					
3720	024316	000							

```

3721 024317 045 047045 020117 NOCLK: .ASCIZ '%NO SYSTEM CLOCK---RESTART @ 1000 OR HIT CONTINUE FOR NO CLOCK%'
3722 024324 054523 052123 046505
3723 024332 041440 047514 045503
3724 024340 026455 051055 051505
3725 024346 040524 052122 040040
3726 024354 030440 030060 020060
3727 024362 051117 044040 052111
3728 024370 041440 047117 044524
3729 024376 052516 020105 047506
3730 024404 020122 047516 041440
3731 024412 047514 045503 000045
3732 024420 051045 046105 041517 RELEERR: .ASCIZ '%RELOCATION ERROR AT '
3733 024426 052101 047511 020116
3734 024434 051105 047522 020122
3735 024442 052101 000040
3736 024446 051045 046105 041517 RELMSG: .ASCIZ '%RELOCATED TO '
3737 024454 052101 042105 052040
3738 024462 020117 000
3739 024465 045 054523 052123 SYSSIZ: .ASCIZ '%SYSTEM SIZE:'
3740 024472 046505 051440 055111
3741 024500 035105 000
3742 024503 113 000
3743 024505 045 051127 052111 KEY: .ASCIZ '%K'
ROTEMB: .ASCIZ '%WRITE BUFFER ROTATION ENABLED, RANGE: '
3744 024512 020105 052502 043106
3745 024520 051105 051040 052117
3746 024526 052101 047511 020116
3747 024534 047105 041101 042514
3748 024542 027104 051040 047101
3749 024550 042507 020072 000
3750
3751 024555 045 051127 052111 ROTDIS: .ASCIZ '%WRITE BUFFER ROTATION DISABLED'
3752 024562 020105 052502 043106
3753 024570 051105 051040 052117
3754 024576 052101 047511 020116
3755 024604 044504 040523 046102
3756 024612 042105 000
3757 024615 045 052113 030461 KTONM: .ASCIZ '%KT11 ENABLED'
3758 024622 042440 040516 046102
3759 024630 042105 000
3760 024633 045 053523 052111 NOSWR: .ASCIZ '%SWITCH REGISTER AT LOC. 1004%'
3761 024640 044103 051040 043505
3762 024646 051511 042524 020122
3763 024654 052101 046040 041517
3764 024662 020056 030061 032060
3765 024670 000045
3766 .EVEN
    
```

```

3767 ;ROUTINE TO LOAD WORST CASE UNIBUS PATTERN IN ALL FREE MEMORY
3768
3769 024672 004467 170042 LOADWC: JSR R4,SAV04
3770 024676 016702 154322 MOV LCDR12,R2 ;SETUP 12 BIT STARTING ADDRESS
3771 024702 056702 154334 ADD OFFSET,R2
3772 024706 026702 154314 CMP HCDR12,R2 ;ANY MEMORY TO LOAD ?
3773 024710 001425 BEQ 88 ;NO, GET OUT
3774 024714 016700 154300 MOV LOCCORE,R0 ;IF NO KT11, R0 WILL CONTAIN PHYSICAL ADDRESS
3775 024720 012704 025006 28: MOV #WCASE,R4 ;LOAD ADDRESS OF WORST CASE DATA TABLE
3776 024724 105767 154354 38: TSTB KTRPRES ;USF KT11?
3777 024730 001404 BEQ 56 ;NO- BRANCH
3778 024732 012700 040000 48: MOV #40000,R0 ;SET VIRTUAL ADDRESS TO THE START OF PAGE 2
3779 024736 010267 145402 MOV R2,KIPAR2 ;MAP PAGE 2 TO PHYSICAL ADDRESS
3780 024742 012703 000400 58: MOV #256,,R3 ;SET UP COUNTER
3781 024746 011420 68: MOV (R4),(R0)+ ;LOAD DATA
3782 024750 015303 DEC R3
3783 024754 00375 BNE 68
3784 024758 062702 000010 ADD #10,R2 ;AFTER EACH 256 WORDS, REMAP
3785 024760 020267 154242 CMP R2,HCDR12 ;DONE?
3786 024764 002403 BLT 78 ;NO- BRANCH
3787 024768 004767 167760 88: JSR PC,PST04
3788 024772 000207 RTS PC ;YES- EXIT
3789 024774 000734 78: TST (R4)+ ;MOVE DATA POINTER
3790 024776 000421 025106 CMP R4,#WCASEE ;TO NEXT WORST CASE VALUE
3791 025002 100258 BLO 38 ;AND CONTINUE
3792 025004 000745 BR 28
3793
3794 025006 177776 000001 177775 WCASE: .WORD 177776,1,177775,2,177773,4,177767,10
3795 025014 000002 177773 000004
3796 025022 177767 000010
3797 025026 177757 000020 177737 .WORD 177757,20,177737,40,177677,100,177577,200
3798 025034 000040 177677 000100
3799 025042 177577 000200
3800 025046 177377 000400 176777 .WORD 177377,400,176777,1000,175777,2000,173777,4000
3801 025054 001000 175777 002000
3802 025062 173777 004000
3803 025066 167777 010000 157777 .WORD 167777,10000,157777,20000,137777,40000,77777,100000
3804 025074 020000 137777 040000
3805 025102 077777 100000
3806 025106 WCASEE:
    
```

```

3807 ;COMMAND TABLE, KEYBOARD COMMANDS ARE MATCHED AGAINST IT.
3808 ;THE MATCH MUST BE EXACT, NO ABBREVIATIONS ARE ALLOWED.
3809
3810 025106 COMTAB;
3811 025106 TOKN  "RUNL",RUNL
3812 025106 .ASCIZ %RUNL%
3813 025114 052522 046116 000 .WORD  RUNL
3814 025116 TOKN  "RUN",RUN
3815 025116 .ASCIZ %RUN%
3816 025124 052522 000116 .WORD  RUN
3817 025126 TOKN  "MOD",MOD
3818 025126 .ASCIZ %MOD%
3819 025134 013222 .WORD  MOD
3820 025136 TOKN  "SEL",SEL
3821 025136 .ASCIZ %SEL%
3822 025144 012462 .WORD  SEL
3823 025146 TOKN  "DES",DES
3824 025146 .ASCIZ %DES%
3825 025154 012500 .WORD  DES
3826 025156 TOKN  "MAP",MAP
3827 025156 .ASCIZ %MAP%
3828 025164 012314 .WORD  MAP
3829 025166 TOKN  "FILL",FILL
3830 025166 .ASCIZ %FILL%
3831 025174 044506 046114 000 .WORD  FILL
3832 025176 TOKN  "KTOFF",KTOFF
3833 025176 .ASCIZ %KTOFF%
3834 025206 012562 043117 000106 .WORD  KTOFF
3835 025210 TOKN  "KTON",KTON
3836 025210 .ASCIZ %KTON%
3837 025216 012632 .WORD  KTON
3838 025220 TOKN  "PON",PON
3839 025220 .ASCIZ %PON%
3840 025226 013004 .WORD  PON
3841 025230 TOKN  "POFF",POFF
3842 025230 .ASCIZ %POFF%
3843 025236 013044 .WORD  POFF
3844 025240 TOKN  "ROTON",ROTON
3845 025240 .ASCIZ %ROTON%
3846 025250 012744 047524 000116 .WORD  ROTON
3847 025252 TOKN  "ROTOFF",ROTOFF
3848 025252 047522 043106 .ASCIZ %ROTOFF%
3849 025260 000 .WORD  ROTOFF
3850 025262 012760 .WORD  ROTOFF
3851 025264 TOKN  "LPON",LPON
3852 025264 .ASCIZ %LPON%
3853 025272 050114 047117 000 .WORD  LPON
3854 025274 TOKN  "CON",CON
3855 025274 .ASCIZ %CON%
3856 025302 013130 .WORD  CON
3857 025304 TOKN  "COFF",COFF
3858 025304 .ASCIZ %COFF%
3859 025312 013170 .WORD  COFF
3860 025314 015 000 .BYTE  15,0 ;CR
3861 025316 004 000 .BYTE  4,0
3862 025320 010646 .WORD  COMCO2
    
```

```

3863 025322 000000 000000 000000 .WORD  0,0,0
    
```

3864 IMPURE MESSAGES.  
 3865  
 3866 025330 050045 051101 052111 PERR1: ,ASCII %PARITY ERROR  
 3867 025336 020131 051105 047522  
 3868 025344 020122 020040  
 3869 025350 020040 020040 PREG: ,ASCII /  
 3870 025356 057  
 3871 025357 040 020040 020040 PCONT: ,ASCII  
 3872 025364 040  
 3873 025365 040 020040 052522 RUNDIA: ,ASCIZ RUN PARITY/MEMORY DIAGNOSTICS%  
 3874 025372 020116 040520 044522  
 3875 025400 054524 046457 046505  
 3876 025406 051117 020131 044504  
 3877 025414 043501 047516 052123  
 3878 025422 041511 022523 000  
 3879 025427 045 040520 044522 PERR2: ,ASCIZ %PARITY ERROR --- (JAM,SPV,PBA,CUA,FLG/INT,WHAMI,CDATA,CTAG/CPU)%  
 3880 025434 054524 042440 051122  
 3881 025442 051117 026440 026455  
 3882 025450 024040 040512 026115  
 3883 025456 051123 026126 041120  
 3884 025464 026101 052503 026101  
 3885 025472 046106 027507 047111  
 3886 025500 026124 034127 046501  
 3887 025506 026111 042103 052101  
 3888 025514 026101 052103 043501  
 3889 025522 041457 052520 022451  
 3890 025530 000  
 3891 025531 045 054523 052123 NUMSYS: ,ASCIZ %SYSTEM ERRORS:  
 3892 025536 046505 042440 051122  
 3893 025544 051117 035123 000  
 3894 025551 040 020040 020040 NUMPWR: ,ASCIZ POWER FAILS:  
 3895 025556 047520 042527 020122  
 3896 025564 040506 046111 035123  
 3897 025572 000  
 3898 025573 045 020040 020040 AMODNM: ,ASCII % AT  
 3899 025600 020040 052101 040  
 3900 025605 040 020040 020040 APC: ,ASCII STAT  
 3901 025612 020040 052123 052101  
 3902 025620 040  
 3903 025621 040 020040 020040 AMDSTA: ,ASCII  
 3904 025626 040  
 3905 025627 040 050040 051501 MDIRE: ,ASCII PASCNT  
 3906 025634 047103 020124  
 3907 025640 020040 020040 027040 APSCNT: ,ASCII ERRCNT  
 3908 025646 020040 051105 041522  
 3909 025654 052116 040  
 3910 025657 040 020040 020040 AERRS: ,ASCIZ  
 3911 025664 000056  
 3912 025667  
 3913 025667 040 020040 020040 ,ODD HOLDS: ,ASCIZ  
 3914 025674 000  
 3915 025675 040 020040 020040 HOLD6: ,ASCIZ  
 3916 025702 020040 000040  
 3917 025706 020040 020040 020040 HOLD8: ,ASCIZ  
 3918 025714 020040 020040 000  
 3919 025721 040 020040 020040 AEND: ,ASCII PC-

3920 025726 020040 041520 055  
 3921 025733 040 020040 020040 AEND1: ,ASCII APC-  
 3922 025740 020040 050101 026503  
 3923 025746 020040 020040 020040 AEND2: ,ASCIZ  
 3924 025754 000040  
 3925 025756 020040 040520 051523 \$PASS: ,ASCII PASS:  
 3926 025764 020043  
 3927 025766 020040 020040 027040 APASS: ,ASCIZ  
 3928 025774 000  
 3929 025775 040 051104 050117 MODEND: ,ASCIZ DROPPED%  
 3930 026002 042520 022504 000  
 3931 026007 105 042116 040520 APSEND: ,ASCII ENDPAS-  
 3932 026014 026523  
 3933 026016 020040 020040 027040 BPCNT: ,ASCIZ  
 3934 026024 000  
 3935 026025 045 020040 020040 TIMX: ,ASCIZ % IS HUNG, MODULE DROPPED,%  
 3936 026032 020040 051511 044040  
 3937 026040 047125 027107 046440  
 3938 026046 042117 046125 020105  
 3939 026054 051104 050117 042520  
 3940 026062 027104 000045  
 3941 026066 051040 047125 044524 TOTIM: ,ASCII RUNTIME-  
 3942 026074 042515 055  
 3943 026077 040 020040 072 THRS: ,ASCII  
 3944 026103 040 035040 TMIN: ,ASCII  
 3945 026106 020040 000 TSECS: ,ASCIZ  
 3946 026111 040 051520 044524 PSTIM: ,ASCII PSTIME-  
 3947 026116 042515 055  
 3948 026121 040 035040 MMINS: ,ASCII  
 3949 026124 020040 000045 MSECS: ,ASCIZ  
 3950 026130 041445 051123 020101 AERROR: ,ASCII %CSRA  
 3951 026136 020040 020040 020040 ACSRAC: ,ASCII CSRC  
 3952 026144 041440 051123 020103  
 3953 026152 020040 020040 020040 ACSRC: ,ASCII STATC  
 3954 026160 051440 040524 041524  
 3955 026166 040  
 3956 026167 040 020040 020040 ASTATC: ,ASCIZ  
 3957 026174 020040 000  
 3958 026177 045 051503 040522 ADTERR: ,ASCII %CSRA  
 3959 026204 040  
 3960 026205 040 020040 020040 ADTE6: ,ASCII S/B  
 3961 026212 020040 027523 020102  
 3962 026220 020040 020040 020040 ADTE3: ,ASCII WAS  
 3963 026226 053440 051501 040  
 3964 026233 040 020040 020040 ADTE2: ,ASCII WRADR  
 3965 026240 020040 051127 042101  
 3966 026246 020122  
 3967 026250 020040 020040 020040 ADTE5: ,ASCII RDADR  
 3968 026256 051040 040504 051104  
 3969 026264 040  
 3970 026265 040 020040 020040 ADTE4: ,ASCIZ  
 3971 026272 020040 000  
 3972 026275 040 040504 040524 ADTE4A: ,ASCIZ DATA ERROR  
 3973 026302 042440 051122 051117  
 3974 026310 000  
 3975 026311 127 051117 021504 ADTEX: ,ASCII WORD:

```

3976 026316 040
3977 026317 040 020040 020040 ADTE7: .ASCIZ ' ,%'
3978 026324 022456 000
3979 026327 040 051105 021522 ERRNMB: .ASCII ' ERR# '
3980 026334 040
3981 026335 040 020040 020040 AERNMB: .ASCIZ ' , '
3982 026342 000056
3983 026344 046445 046505 043040 MERR1: .ASCII '%MEM FAILURE= PA '
3984 026352 044501 052514 042522
3985 026360 020055 040520 040
3986 026365 040 020040 020040 MERR2: .ASCII ' S/B '
3987 026372 020040 027523 020102
3988 026400 020040 020040 020040 MERR3: .ASCII ' WAS '
3989 026406 053440 051501 040
3990 026413 040 020040 020040 MERR4: .ASCIZ ' '
3991 026420 020040 000
3992 026423 045 020040 020040 AERNM: .ASCII '% HAD '
3993 026430 020040 040500 020104
3994 026436 020040 020040 027040 AERCT: .ASCII ' . DATA ERRORS OUT OF '
3995 026444 042040 052101 020101
3996 026452 051105 047522 051522
3997 026460 047440 052125 047440
3998 026466 020106
3999 026470 020040 020040 027040 AERTOT: .ASCIZ ' . WORDS READ%'
4000 026476 053440 051117 051504
4001 026504 051040 040505 022504
4002 026512 000
4003 026513 040 040 040 DECVAL: .BYTE 40,40,40,40,40,40
4004 026516 040 040 040
    
```

```

4005 ;BUFFER AREAS,
4006
4007 ;EVEN
4008 026522
4009 026712 MODQ: ;ROOM FOR 60 MODULE POINTERS,
4010 026752 000014 KBUF: ;KEYBOARD BUFFER, KRUF1 LONG
4011 027002 000004
4012 027012
4013 027012 000100
4014 027212
4015 027212 000020
4016 027252
4017 027252 000620
4018 030072 001130
4019
4020 031222 000000
4021 031224 000000
4022 031226 000000
4023 031230 000000
4024 031232 000000
4025 031234 000000
4026 031236 000000
4027 031240 000000
4028 031242 000000
4029 031244 000000
4030 031246 000000
4031 031250 000000
4032 031252 000000
4033 031254 000000
4034 031256 000000
4035 031260 000000
4036 031262 000000
4037 031264 000000
4038 031266 000000
4039 031270 000000
4040 031272 000000
4041 031274 000000
4042 031276 000000
4043 031300 000000
4044 031302 000000
4045 031304 000000
4046 031306 000000
4047 031310 000000
4048 031312 000000
4049 031314 000000
4050 031316 000000
4051 031320 000000
4052 031322 000000
4053 031324 000000
4054 031326 000000
4055 031330 000000
4056 031332 000000
4057 031334 000000
4058 031336 000000
4059 031340 000000
4060 031342 000000
    
```

```

4061 031344 000000 OPEN ;PATCH AREA
4062 031346 000000 OPEN ;PATCH AREA
4063 031350 000000 OPEN ;PATCH AREA
4064 031352 000000 OPEN ;PATCH AREA
4065 031354 000000 OPEN ;PATCH AREA
4066 031356 000000 OPEN ;PATCH AREA
4067 031360 000000 OPEN ;PATCH AREA
4068 031362 000000 OPEN ;PATCH AREA
4069 031364 000000 OPEN ;PATCH AREA
4070 031366 000000 OPEN ;PATCH AREA
4071 031370 000000 OPEN ;PATCH AREA
4072 031372 000000 OPEN ;PATCH AREA
4073 031374 000000 OPEN ;PATCH AREA
4074 031376 000000 OPEN ;PATCH AREA
4075 031400 000000 OPEN ;PATCH AREA
4076 031402 000000 OPEN ;PATCH AREA
4077 031404 000000 OPEN ;PATCH AREA
4078 031406 000000 OPEN ;PATCH AREA
4079 031410 000000 OPEN ;PATCH AREA
4080 031412 000000 OPEN ;PATCH AREA
4081 031414 000000 OPEN ;PATCH AREA
4082 031416 000000 OPEN ;PATCH AREA
4083 031420 000000 OPEN ;PATCH AREA
4084
4085 031422 000200 LOGRUF: ,BLKW 200 ;BUFFER HOLDS LAST COPY TAKEN OF 11/60 LOG
4086
4087 032022 .*. ;FIRST ADDRESS AFTER QUEUES
4088 024037 RUF5IZ=AMODNM-START
4089
    
```

```

4090 ;ROUTINE TO DETERMINE WHETHER WRITE BUFFER ROTATION SHOULD TAKE PLACE,
4091 ;AND TO DETERMINE CORE LIMITS OF BUFFER ROTATION. ALSO TO DETERMINE USE OF
4092 ;RTI OR RTT INSTRUCTION.
4093
4094 027252 012767 150530 SETBUF: ,=ICQ
4095 027260 005044 MOV #16,RESIV ;SET UP A RESERVED INSTRUCTION TRAP.
4096 027262 012749 027320 CLR -(SP) ;CLEAR A WORD ON THE STACK FOR A NEW PSW
4097 027266 000006 MOV #28,-(SP) ;SET UP TO EXIT WITH RTI INSTRUCTION.
4098 027270 012767 000002 156410 18: RTT ;IF RTT NOT VALID IT WILL TRAP OUT.
4099 027276 012767 000002 153632 MOV #RTI,TRCIA ;TRAP COMES HERE. CHANGE RTT'S TO RTI'S.
4100 027304 016767 152060 152066 MOV #RTI,RTTI ; BY SETTING THE INSTRUCTION VALUE
4101 027312 016767 152054 152062 MOV CHK2,CHK1 ;CHANGE SET OF CHKSUM WORDS BECAUSE
4102 ;OF RTI'S REPLACING RTT'S
4103 027320 012767 150462 28: MOV #PESINT,RESIV ;OF ALL RTI'S TO THOSE OF RTI'S
4104 ;RESTORE THE RESERVED INSTRUCTION TRAP VECTOR
4105 027326 105067 152001 CLR #MDI45 ;CLEAR THE PDP-11/45 INDICATOR
4106 027332 012767 150444 MOV #38,BUSEV ;SET UP A NON-EXISTENT ADDRESS TRAP
4107 027340 005737 177772 TST #PIRQ ;IS THE PROGRAMMED INTERRUPT REQUEST REGISTER THERE?
4108 027344 112767 000001 151761 MOV #1,MDL45 ;IF IT IS, SET THE FLAG INDICATING AN 11/45
4109 027352 105067 151726 CLR KTPRES ;CLEAR THE MEMORY MANAGEMENT INDICATOR
4110 027356 012767 027402 150420 MOV #48,BUSEV ;SET UP A NON-EXISTENT ADDRESS TRAP
4111 027364 005037 177572 CLR #SSRO ;IS THERE A KT-11 ON THIS SYSTEM?
4112 027370 105167 151710 COMB KTPRES ;IF THERE IS, SET THE KT11 PRESENT INDICATOR
4113 027374 115767 151704 151670 MOV #KTPRES,KTSAV ;SAVE FOR USE IN SECOND PASS OF CHAINING
4114 027402 012767 015246 150374 48: MOV #BUSERR,BUSEV ;REPLACE THE BUS ERROR TRAP VECTOR
4115 027410 005067 150362 CLR PSW ;RESET THE PSW
4116 027414 105067 151673 STBFA: CLR RTI ;ASSUME NO BUFFER ROTATION.
4117 027420 105067 151671 CLR RELOC ;ASSUME NO CODE RELOCATION
4118 027424 016767 150350 151756 MOV #7,LOCORE ;GET LOCORE ADDR. FROM LOC 0
4119 027432 001004 BNE 18 ;BP IF LINKED WITH CONFIG/LINKER PROGRAM
4120 ;IF NOT ASSUMES NOT IN CHAIN MODE
4121 027434 005167 151560 COM LOCORE ;INDICATE SETBUF HAS BEEN RUN
4122 027440 000167 000420 JMP SETXT ;GO PREPARE TO EXIT SETBUF ROUTINE
4123 027444 004567 165152 18: JSR #5,TOP12S ;GET TOP 12 BITS OF LOCORE PHYS ADDR
4124 027450 001220 LOCORE ;AND PUT IT INTO LCOPI2
4125 027452 032767 000777 151540 BIT #777,LOCORE ;IS IT ON A 256 WORD BOUNDARY ?
4126 027460 031414 BEQ 26 ;YES, CONTINUE
4127 027462 042767 000777 151530 BIC #777,LOCORE ;NO, CLEAR OUT LOWER BITS
4128 027470 042767 000007 151526 BIC #7,LCOPI2 ;ROUNDING OFF TO NEXT HIGHEST 256 BOUNDARY
4129 027476 062767 001000 151514 ADD #1000,LOCORE ;ROUND UP TO THE NEXT 256 WORD BOUNDARY
4130 027504 062767 000010 151512 ADD #10,LCOPI2 ;ALSO INDICATE THAT BOUNDARY IN LCOPI2
4131 027512 016767 150324 151506 28: MOV #42,HCOR12 ;ARE WE IN CHAIN MODE?
4132 027520 001403 BEQ KTEST ;IF NO, GO DETERMINE IF KT IS TO BE USED
4133 027522 112767 177777 151562 MOV #1,CHN ;IF YES, SET THE CHAIN INDICATOR
4134 027530 105767 151550 KTEST: TSTB KTPRES ;IS THE KT11 AVAILABLE FOR USE?
4135 027534 001035 BNE SETKI ;IF YES, GO SIZE MEMORY USING IT
4136 027536 012767 027600 150240 MOV #58,4 ;IF NO, ONLY SIZE MEMORY UP TO 28K
4137 027544 012767 174000 MOV #4000,RO ;INITIALIZE LOOP COUNTER TO POINT TO BANK 0
4138 027550 005067 CLR R1 ;INITIALIZE THE COUNTER OF 1K CHUNKS(BANKS)
4139 027552 027600 004000 18: ADD #4000,RO ;POINT TO THE FIRST WORD IN THE NEXT 1K BANK
4140 027556 027600 005000 ADD #0,(RO) ;WILL TRAP IF LOC DOESN'T EXIST
4141 027562 042767 INC R1 ;IF NO, CONTINUE PROCESSING AT 48
4142 ;IF YES, UP THE MEMORY BANK COUNT
4143 027564 027600 000034 CHP #28,R1 ;HAVE 28 BANKS BEEN COUNTED?
4144 027570 001270 BNE 48 ;IF NO, GO CONTINUE SIZING
4145 027572 062767 004000 ADD #4000,RO ;IF YES, INDICATE THE NUMBER OF WORDS IN RO
    
```

```

4146 027576 000401          BR      68          ;SKIP OVER THE NEXT INSTRUCTION
4147 027600 022626          58:    CMP      (SP)+,(SP)+          ;IF THERE WAS NO LOCATION AT LAST TRY,
4148                                     ;REMOVE THE TRAP FROM THE STACK
4149 027602 000300          68:    SWAB     RO          ;ISOLATE THE 12 HIGH ORDER BITS, FIRST,
4150                                     ;GET THE MOST SIGNIFICANT BITS IN THE LOW BYTE
4151 027604 006300          ASL     RO          ;ALIGN THE BITS TO ACCOUNT FOR
4152 027606 006300          ASL     RO          ;EXTENDED ADDRESS BITS (17 + 18)
4153 027610 042700 170037          BIC     #170037,RO          ;ELIMINATE THE UNWANTED BITS FROM RO
4154 027614 105767 151472          TSTB   CHN          ;ACT11 CHAIN?
4155 027620 100054          BPL     SETCOM          ;NO- DON'T ADD ROOM FOR MONITOR
4156 027622 162700 000100          SUB     #100,RO          ;YES- ALLOW ROOM FOR MONITOR
4157 027626 000451          BR      SETCOM          ;GO TEST THE MEMORY
4158 027630 004767 162160          SETKT: JSR     PC,KTINIT          ;TURN ON THE KT11
4159 027634 012737 077406 172304          MOV     #77406,##KIPDR2
4160 027642 012737 000001 177572          MOV     #1,##SSRO          ;TURN ON KT11
4161
4162 027650 012737 027722 000004          MOV     #28,##4          ;SETUP TIMEOUT VECTOR
4163 027656 016701 151342          MOV     LCOR12,R1          ;GET THE TOP OF THE PROGRAM
4164 027662 062701 000040          ADD     #40,R1          ;ROUND IT UP
4165 027666 042701 000037          BIC     #37,R1          ;MAKE SURE IT'S A 1K BOUNDARY
4166 027672 010137 172344          MOV     R1,##KIPAR2          ;LOAD THE KT11 REGISTER
4167 027676 062737 000000 040000 18:    ADD     #0,##40000          ;TIME OUT IF THIS LOCATION IS NXH
4168 027704 062737 000040 172344          ADD     #40,##KIPAR2
4169 027712 001371          BNE     18
4170 027714 012700 007600          MOV     #7600,RO
4171 027720 000403          BR      18
4172 027722 022626          28:    CMP      (SP)+,(SP)+
4173 027724 016700 142414          MOV     KIPAR2,RO
4174 027730 105767 151356          38:    TSTB   CHN          ;CHAIN MODE?
4175 027734 100006          BPL     SETCOM          ;NO, JUST GO TEST MEMORY
4176 027736 005067 151342          CLR     KTPRES          ;YES SHUT OFF KT FOR FIRST CHAIN PASS
4177 027742 010067 151326          MOV     RO,HCR12A          ;SAVE THE REAL MEMORY SIZE
4178 027746 000005          RESET          ;TURN KT OFF
4179 027750 000667          BR      KTTST          ;NOW GO SIZE WITHOUT IT
4180 027752 012767 015246 150024 SETCOM: MOV     #BUSERR,4          ;RESET BUS ERROR TRAP.
4181 027760 010067 151242          MOV     RO,HCR12          ;HI ADDR TO HCR12.
4182 027764 004767 000102          SETCM1: JSR     PC,TSTMEM          ;TEST MEMORY FOR BASIC FUNCTIONING
4183
4184 027770 004767 000626          JSR     PC,ANYPAR          ;VIA ADDRESS UP/DOWN TEST
4185 027774 016700 151226          MOV     HCR12,RO          ;CHECK FOR PARITY MEN. AND TURN IT ON
4186 030000 010067 151224          MOV     RO,HCRORSV          ;DIFF BETWEEN LCR12 AND HCR12
4187 030004 166700 151214          SUB     LCR12,RO          ;INITIALIZE SAVE LOCATION FOR KTON/OFF COMMANDS
4188 030010 022700 000040          CMP     #40,RO          ;MUST BE AT LEAST 40
4189 030014 101023          BHI     18              ;IS IT?
4190 030016 105267 151271          BHI     SETXT          ;FORGET IT IF NOT.
4191 030022 105767 151256          INCB   ROTI          ;ALLOW ROTATION.
4192 030026 001413          TSTB   KTPRES          ;NO RELOCATION IF NO KT11
4193 030030 022700 000400          BEQ     18
4194 030034 101010          CMP     #400,RO          ;IF 8K OR GREATER, ALLOW CODE RELOCATION
4195 030036 016767 151164 151174          BHI     18              ;BRANCH IF NOT
4196 030044 166767 151154 151166          MOV     HCR12,OFFMX          ;CALCULATE THE HIGHEST LOC TO WHICH
4197 030052 105267 151237          SUB     LCR12,OFFMX          ;THE PROGRAM CAN BE RELOCATED
4198 030056 004767 174610          INCR   RFL0C          ;ENABLE RELOCATION
4199 030062 000005          JSR     PC,LOADMC          ;LOAD WORST CASE UNIBUS NOISE PATTERN
4200 030064          RESET          ;TURN OFF KT11 IF IN USE
4201 030064 005067 147710          SETXT: CLP     0          ;CLEAR LOC 0.
    
```

```

4202 030070 000205          PTS     P5          ;EXIT.
    
```

ADDRESS	MEMORY	ADDRESS	UP/DOWN	TEST
4203				
4204				
4205	030072	026707	151130	151124 TSTMEN: CMP HCOIP2,LCOR12 ;ANY MEMORY TO TEST ?
4206	030100	001001		BNE 58 ;YES, CONTINUE
4207	030102	000207		RTS PC ;NO, RETURN
4208	030104	016702	151114	58: MOV LCOR12,R2 ;LOAD 12 BIT LOCORE ADDRESS
4209	030110	016701	151104	MOV LOCORE,R1 ;R1 CONTAINS LOCORE PHYS ADDRESS
4210	030114	012767	030476	147662 MOV #NOMEM,4 ;LOAD TIME-OUT VECTOR
4211	030122	105767	151156	18: TSTB KTPRES ;USE KT11?
4212	030126	001002		BNE 28 ;YES- BRANCH
4213	030130	010100		MOV R1,R0 ;NO- SET VA EQUAL TO PA
4214	030132	000404		BR 38
4215	030134	012700	040000	28: MOV #40000,R0 ;SET VA TO PAGE 2
4216	030140	010267	142200	MOV R2,KIPAP2 ;MAP PAGE 2 TO PHYSICAL ADDRESS
4217	030144	012703	000040	38: MOV #32,,R3 ;COUNT 1 BLOCK
4218	030150	010120		48: MOV R1,(R0)+ ;LOAD LOCATION WITH ITS ADDRESS
4219	030152	062701	000002	ADD #2,R1
4220	030156	005303		DEC R3
4221	030160	001373		BNE 46
4222	030162	005202		INC R2 ;AFTER 1 BLOCK, UP PAGE ADDRESS FIELD
4223	030164	020267	151036	CMP R2,HCOIP2 ;AT TOP?
4224	030170	002754		BLT 18 ;NO- CONTINUE
4225	030172	005302		DEC R2 ;YES- TEST GOING DOWN
4226	030174	162701	000002	SUB #2,R1
4227	030200	105767	151100	TSTM1: TSTB KTPRES ;USE KT11?
4228	030204	001002		RNE 18 ;YES- BRANCH
4229	030206	010100		MOV P1,R0 ;NO- LOAD VIRTUAL ADDRESS POINTER WITH PHYS ADDRESS
4230	030210	000404		BR 28
4231	030212	012700	040076	18: MOV #40076,R0 ;MAP VA TO TOP OF 1ST BLOCK IN PAGE 2
4232	030216	010267	142122	MOV R2,KIPAP2 ;MAP PAGE 2 TO PA
4233	030222	012703	000040	28: MOV #32,,R3 ;COUNT 1 BLOCK
4234	030226	011067	150764	38: MOV #R0,MEMWAS
4235	030232	020167	150760	CMP R1,MEMWAS ;DOES LOCATION CONTAIN ITS PA?
4236	030236	001120		RNE MERR ;NO- BRANCH
4237	030240	162700	000002	48: SUB #2,R0
4238	030244	162701	000002	SUB #2,R1
4239	030250	005303		DEC R3
4240	030252	001365		RNE 36
4241	030254	005302		DEC P2 ;AFTER CHECKING 1 BLOCK, ADDRESS NEXT
4242	030256	020267	150742	CMP R2,LCOR12 ;DONE?
4243	030262	002346		BGE TSTM1 ;NO- CONTINUE CHECKING
4244	030264	016702	150736	MOV HCOIP2,R2 ;YES- SET TO LOAD MEMORY
4245	030270	010201		MOV R2,R1 ;FROM TOP DOWN
4246	030272	005302		DEC R2
4247	030274	006301		ASL R1
4248	030276	006301		ASL R1
4249	030300	006301		ASL R1
4250	030302	006301		ASL R1
4251	030304	006301		ASL R1
4252	030306	005301		DEC P1
4253	030310	006301		ASL R1
4254	030312	105767	150766	TSTM2: TSTB KTPRES ;USE KT11?
4255	030316	001002		BNE 18 ;YES- BRANCH
4256	030320	010100		MOV P1,R0 ;NO- LOAD VIRTUAL POINTER WITH PHYSICAL ADDRESS
4257				
4258	030322	000404		RR 28

4259	030324	012700	040076	18: MOV #40076,R0 ;LOAD VIRTUAL WITH TOP ADDRESS OF 1ST BLOCK IN PAGE 2
4260				
4261	030330	010267	142110	MOV R2,KIPAP2 ;MAP PAGE 2 TO PHYSICAL
4262	030334	012703	000040	38: MOV #32,,R3 ;COUNT 1 BLOCK
4263	030340	010110		MOV R1,R0 ;LOAD LOCATION WITH ITS PHYS ADDRESS
4264	030342	005110		COM #R0 ;COMPLEMENT IT
4265	030344	162701	000002	SUB #2,R1
4266	030350	162700	000002	SUB #2,R0
4267	030354	005303		DEC R3
4268	030356	001370		RNE 36
4269	030360	005302		DEC P2 ;AFTER LOADING A BLOCK, SET UP FOR NEXT
4270	030362	020267	150636	CMP R2,LCOR12 ;DONE?
4271	030366	002351		BGE TSTM2 ;NO- BRANCH
4272	030370	005202		INC R2 ;YES- SETUP TO CHECK IT
4273	030372	062701	000002	ADD #2,R1
4274	030376	010104		MOV R1,R4
4275	030400	105767	150700	TSTM3: TSTB KTPRES ;USE KT11?
4276	030404	001002		BNE 18 ;YES- BRANCH
4277	030406	010400		MOV R4,R0 ;SET VIRTUAL ADDRESS EQUAL TO PHYS ADDRESS
4278	030410	000404		BR 28
4279	030412	012700	040000	18: MOV #40000,R0 ;SET VIRTUAL ADDRESS TO START OF PAGE 2
4280	030416	010267	141722	MOV R2,KIPAP2 ;MAP PAGE 2 TO PHYSICAL ADDRESS
4281	030422	012703	000040	28: MOV #32,,R3 ;COUNT A BLOCK
4282	030426	011067	150564	38: MOV #R0,MEMWAS
4283	030432	010401		MOV R4,R1
4284	030434	005101		COM R1
4285	030436	020167	150554	CMP R1,MEMWAS ;DOES LOCATION CONTAIN COMPLEMENT OF ITS PHYS ADDRESS?
4286	030442	001016		BNE MERR ;NO- BRANCH
4287	030444	005720		TST (R0)+
4288	030446	062704	000002	ADD #2,R4
4289	030452	005303		DEC R3
4290	030454	001364		BNE 38
4291	030456	005202		INC R2 ;CHANGE 12 BIT ADDRESS AFTER CHECKING A BLOCK
4292	030460	020267	150542	CMP R2,HCOIP2 ;DONE?
4293	030464	002745		BLT TSTM3 ;NO- BRANCH
4294	030466	012767	015246	147310 MOV #RUSERR,4 ;RESET THE TIME-OUT TRAP VECTOR
4295	030474	000207		RTS PC ;YES- RETURN
4296				
4297	030476	022626		NOMEM: CMP (SP)+,(SP)+ ;POP THE STACK POINTER TWICE
4298	030500	012767	015246	147276 MERR: MOV #RUSERR,4 ;RESET THE TIME-OUT TRAP VECTOR
4299	030506	105267	150623	INCB MEMERR
4300	030512	010067	150466	MOV R0,MEMVA ;SAVE FAILING ADDRESS
4301	030516	104413	001204	GETPAS, MEMVA
4302	030522	010167	150466	MOV R1,MEMSBE ;SAVE EXPECTED CONTENTS
4303	030526	016746	150454	MOV MEMPA,-(SP) ;CONVERT PHYSICAL ADDRESS TO ASCII
4304	030532	016746	150452	MOV MEMEA,-(SP)
4305	030536	004567	163410	JSR R5,DACNVX
4306	030542	026365		MERR2
4307	030544	016746	150444	MOV MEMSBE,-(SP) ;CONVERT SHOULD BE TO ASCII
4308	030550	004567	163364	JSR R5,DACNV
4309	030554	026400		MERR3
4310	030556	016746	150434	MOV MEMWAS,-(SP) ;CONVERT WAS VALUE TO ASCII
4311	030562	004567	163352	JSR R5,DACNV
4312	030566	026443		MERR4
4313	030570	004567	164026	JSR R5,TOPI2S ;GET NEW HIGH CORE VALUE
4314	030574	001206		MEMPA

4315	030576	016767	150410	150422	MOV	MEM12,HCOR12
4316	030604	042767	000007	150414	BIC	#7,HCOR12
4317	030612	162767	000040	150406	SUB	#40,HCOR12
4318	030620	000207			RTS	PC

```

4319                                     ;THIS ROUTINE CHECKS FOR PARITY MEMORY
4320
4321 030622 105067 150512 ANYPAR: CLR B PPRES ;CLEAR PARITY PRESENT FLAG
4322 030626 105067 150507 CLR B CNT1 ;CLEAR # OF REGISTERS COUNTER
4323 030632 012767 030654 147150 MOV #15,RESIV ;SET UP FOR A RESERVED INSTR. TRAP
4324 030640 076600 000101 WED,PDSEV ;IS THIS AN 11/60 ?
4325 030644 152767 000002 150466 BISR #R11,PPRES ;YES, SET 11/60 PARITY PRESENT FLAG
4326 030652 000401 BR 118 ;GET OUT
4327
4328 030654 022626 18: CMP (SP)+,(SP)+ ;POP STACK POINTER TWICE
4329 030656 012767 015276 147124 118: MOV #RESINT,RESIV ;RESTORE RESERVED INSTR. TRAP VECTOR
4330 030664 012701 172100 MOV #172100,R1 ;LOAD ADDRESS OF 1ST REGISTER
4331 030670 012767 030714 147106 MOV #33,BUSEV ;LOAD TIME-OUT VECTOR
4332 030676 012702 001430 MOV #PARTAB,R2 ;LOAD ADDRESS OF TABLE OF ADR'S
4333 030702 005011 29: CLR (#1) ;IF REGISTER IS THERE - CLEAR IT
4334 030704 105267 150431 INCR CNT1 ;IT'S HERE, COUNT IT
4335 030710 010122 MOV R1,(R2)+ ;PUT IT'S ADDRESS INTO THE TABLE
4336 030712 000401 BR 48 ;CONTINUE
4337
4338 030714 022626 38: CMP (SP)+,(SP)+ ;POP THE STACK POINTER TWICE
4339
4340 030716 062701 000002 48: ADD #2,R1 ;STEP TO THE NEXT ADDRESS
4341 030722 022701 172136 CMP #172136,R1 ;ALL REGISTERS CHECKED ?
4342 030726 002365 RGE 28 ;NO, DO THE NEXT ONE
4343 030730 105767 150405 TSTR CNT1 ;YES, FIND ANY PARITY REGISTERS ?
4344 030734 001403 BEQ 58 ;NO, GET OUT
4345 030736 152767 000001 150374 BLSB #BIT0,PPRES ;YES, SET PARITY PRESENT
4346
4347 030744 012767 015246 147032 58: MOV #BUSERR,BUSEV ;RESET TIME-OUT VECTOR
4348 030752 012767 015672 147134 MOV #PARERR,PARVCT ;RESET PARITY VECTOR
4349 030760 000207 RTS PC ;RETURN
4350
4351 000001 .END
    
```







MDDO	026522 G	32#	470	668	1115	1753	1770	1810	1944	1991	2272	3120	3213	3243
		4008#												
MODREA	001124	293#	1550*	1562*										
MODPPA	001122	292#	1569*	1574										
MODR12	001126	294#	1583											
MODWEA	001116	290#	1553*	1565*	1598*	1600								
MODWPA	001114	289#	1570*	1572	1597*	1601								
MODW12	001120	291#	1582											
MORE	010576	1674#												
MOVBNK	001250	341#	3453	3454	3458*									
MOVFLG	001345	309#	3419*	3433*	3532									
MOVING	001325	371#	2603	3425	3430*	3459*	3504*	3510*						
MSCONT	003326	751	754#											
MSECS	026124	3151#	3152*	3949#										
MSGDS =	104422	1635	2571#											
MSGD.	003466	743	790#											
MSGD1.	003556	743	806#											
MSGFLG	001346	743	750*	752*	760									
MSGNS =	104411	2553#												
MSGN.	003322	743	752#											
MSGSS =	104416	2563#												
MSG.S.	003312	743	750#											
MSG.S.	= 104406	505	508	516	520	526	534	1261	1262	1293	1294	1346	1360	1374
		1379	1401	1405	1406	1407	1440	1662	1690	1695	1741	1745	1835	1841
		1968	2009	2023	2027	2043	2057	2059	2067	2081	2093	2101	2143	2151
		2158	2175	2264	2527	2528	2547*	2582	2583	2611	2621	2623	2640	2650
		2676	2704	2849	2850	2854	2885	2886	2887	2907	2908	2930	2931	3034
		3540	3556	3567	3568									
MSG.	003426	743	776#											
MSG1.	003444	759	771	780#										
MVCODE	021474	528	1067	1390	1751	1839	2607	2753	3404#					
MVCCDS	021564	3420#												
MVCCD1	021574	3422#	3519											
MVCCD2	022040	3441	3451	3457#	3496	3503								
MVCCD3	022312	3497	3504#											
MVDOWN	001330	374#	3440*	3450*	3511									
MVERR	022564	3479	3487	3553#										
MVMAP	022412	3434	3518	3521#										
MVMSG	022366	3512	3515#											
MVRTRY	022060	3460#	3566											
MXWBF	001256	344#	1859*	1862*	1863*	1864*	1865*	1866*	1867*	1868*	1870*	2832*	2833*	2834*
		2835#	2836*	2837*	2838*	2840*	3320	3322						
		28	37	152	321	343	823	870	876	883	911	957	977	1160
NEW =	***** II	1627	1780	1962	2537	2541	2549	3573	3766	3907	3956	3979		
NOCLK	024317	3034	3721#											
NOKT	023776	520	2027	3683#										
NOMEH	030476	4210	4297#											
NOPAR	024021	2059	2067	3687#										
NOSWR	024633	508	3760#											
NSTOP	001352	394#	651*	1145	1812*	2769	2804							
NUMBEA	001052	270#	895	901	992									
NUMBER	001046	268#	735*	817*	892*	893	898*	899	990	993				
NUMBPA	001050	269#	894	900	991									
NUMPWR	025551	2930	3894#											
NUMSYS	025531	2907	3891#											
OACNV	014140 G	32#	850	852	888	890	912	914	964	995	1954	1957	2156	2291#

DACNVX	014152	2702	2712	4308	4311									
DACNVR	014160	830	841	848	896	902	908	920	997	2141	2295#	2699	4305	
DASCEA	001112	2149	2298#											
OFFMX	001240	286#	2291*	2300*	2322	2332	2337							
OFFSET	001242 G	336#	1063	1739	2739*	4195*	4196*							
		34#	337#	601	605	1061	1786	1790	1804	1846	1853	2752	2790	2796
		2818	2832	3429*	3431	3436	3438	3448	3455	3491	3505*	3506	3507	3509
		3522	3533	3542	3771									
OFFSTD	001244	338#	527*	1061*	1062*	1063	1065*	1389*	1711*	1747*	1838*	2752*	3409	3416#
		3431	3435											
OFFSTM	001246	339#	342*	3435*	3436	3439	3449	3456	3491	3495	3500	3505		
OPEN =	000000	115#	204	208	209	234	237	238	239	240	241	242	243	244
		245	246	247	248	249	250	251	252	253	254	255	256	258
		260	261	262	264	265	266	267	268	269	270	271	272	273
		274	275	276	277	278	279	280	281	282	283	284	285	286
		289	290	291	292	293	294	295	296	297	298	299	300	301
		302	303	304	305	306	307	321	322	323	324	325	326	327
		329	330	332	334	335	336	337	338	339	341	342	346	347
		348	349	350	356	361	362	363	364	365	366	367	370	371
		372	373	374	375	378	379	380	384	385	386	387	388	389
		390	391	393	394	395	396	419	420	421	422	423	424	425
		426	428	429	430	443	444	448	4020	4021	4022	4023	4024	4025
		4026	4027	4028	4029	4030	4031	4032	4033	4034	4035	4036	4037	4038
		4039	4040	4041	4042	4043	4044	4045	4046	4047	4048	4049	4050	4051
		4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063	4064
		4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077
		4078	4079	4080	4081	4082	4083							
PAENBL	012232	1434	1918#	2056										
PAMSG	015776	1928	2692	2697#										
PAMSG1	016032	2622	2677	2708#										
PAOFF	024034	2081	3689#											
PARERR	015672	222	2510	2666*	4348									
PARTAB	001430	427#	1922	2073	2680	4332								
PARVCT	000114	222#	2510*	2511*	4348*									
PASEND	005310	741	1078#											
PASREL	001033 G	36#	260*	1139	1825*	1826*	1827*	1828*	1831*					
PASTOT	001032	258#	1134*	1139										
PASTYM	020062	1085	3101#											
PCLEAR=	***** G	34#	403											
PCONT	025357	2703	3871#											
PDP70 =	***** II	1	27	28	29	288	369	427	501	518	522	528	829	893
		990	997	1066	1390	1432	1571	1582	1727	1751	1756	1839	1845	1912
		1916	2004	2022	2052	2086	2104	2465	2594	2599	2607	2618	2619	2649
		2652	2664	2751	2753	2779	2790	2796	2798	2818	2832	2871	2994	3351



SHORT = ***** U	1	27	28	29	145	221	288	316	351	395	421	474	510
	522	617	924	1021	1061	1077	1082	1092	1101	1110	1131	1391	1402
	1409	1428	1550	1562	1569	1597	1610	1707	1714	1723	1756	1845	1856
	1872	1997	2045	2084	2439	2500	2510	2581	2599	2603	2619	2627	2642
	2664	2734	2791	2795	2817	2826	2857	2869	2994	3191	3196	3303	3339
	3395	3573	3580	3602	3666	3680	3811	3851	3866	3935	4008	4010	4019
	4104	4157	4158	4184	4319								
SINPUT 010666	1700#	2122											
BLDS 012514	1975	1984#											
BLDBA 012514	1985#												
BLDSB 012530	1721#	1973#	1979#	1989#									
BLDSC 012542	1986	1991#											
BLDSD 012546	1992#	1995											
BLDSE 012552	1722#	1974#	1980#	1994#									
BLDSL 012536	1987	1990#	1993										
SMODE = 040000	93#	1893											
SPBOT 027212	460	498	540	1387	2598	2618	2634	3534	4014#				
SPCFLG 001010	239#	1656	1658#	1692#	1701#								
SPOINT# 000024	156#	622											
SPSAV 001152	304#	701#	1184	1200	1222	3190							
SPSPEC 027252	2593	2608	4016#										
SR 001176 G	35#	318#	463#	465	467#	506	758	802	922	975	979	1086	1595
	3071	3142	3538										
SRETRN 001056	272#	1659	1700#										
SSP 027012	189#	4012#											
SSRO = 177572	78#	482#	497#	515#	1431#	2005#	2015	2020#	2581#	2630#	2644#	2646#	2730#
	2746	2749#	3411#	3413#	4111#	4160#							
SSR3 = 172516	79#												
START 001534	229	231	333	344	413	459#	2757	4088					
START2 002042	514#												
START3 002062	513	520#											
START4 002066	519	522#											
STAT = 000020	153#	597	603	608	618	693	694	697	1020#	1039	1046	1119	1121
	1149	1152	1154#	1642	1781#	1782	1784	1788	1792	1795	1798	1947	1956
	1989#	1994#	3219	3221	3223	3225	3227	3268#	3269				
STAT1 = 000021	154#												
STBFA 027414	4116#												
STOP 001024	251#	581	586	1042	1050	1131	1147#	2771#	2773	2806#	3241		
SUMARY 022753	1401	3584#											
SVRO = 000032	159#	1176	1540										
SVR1 = 000034	160#												
SVR2 = 000036	161#												
SVR3 = 000040	162#												
SVR4 = 000042	163#												
SVR5 = 000044	164#												
SVR6 = 000046	165#	622#	702	1221#	1634#	1644							
SYSBAD 024145	2650	3702#											
SYSCNT 001264	348#	1712#	2602#	2645	2647#	2648	2894						
YSERI 001310	361#	541#											
YSERR 023601	2611	3660#											
YSERP1 023420	2621	3640#											
YSIZ 017052	510	2871#											
YSLIM 001350	392#	2648											
YSUM 017140	1409	2892#											
YSSIZ 024465	2885	3739#											
TABADR 001064	275#	754#	76#	872#	959								

TENPWR 014472	2355	2376#											
TFLAG 001351	393#	3069	3097#	3099#	3101#	3114	3135	3140					
THRS 026077	3079#	3080#	3081#	3943#									
TIMX 026025	3290	3296	3935#										
TIMX1 024230	3280	3711#											
TITLE 022672	505	3573#											
TKB 001164 G	35#	313#	1315	1326									
TKS 001162 G	35#	312#	543#	1314#	1347#	1400#	2484	3420#					
TMINS 026103	3082#	3083#	3944#										
TOP125 014622	1571	1573	2421#	4123	4313								
TOTIM 026066	929	1026	1105	1406	3297	3941#							
TO185 014672	1845	2446#	2857	3541									
TPB 001170 G	35#	315#	1392#	1522#	3304#								
TPA 001166 G	35#	314#	1391#	1465#	1496#	3303#							
TRAPX = 000024	2531#	2535#	2537#	2539#	2541#	2543#	2545#	2547#	2549#	2551#	2553#	2555#	2557#
	2559#	2561#	2563#	2565#	2567#	2569#	2571#	2573#					
TRCI 005700	192	1162#											
TRCIA 005706	1164#	1167	4098#										
TRCIB 005724	1168#												
TRCIC 005710	1163	1165#											
TRCPC 001070	277#	568	716	719#	1168#								
TRCPW 001072	278#	717	1169#										
TRCV 000014	192#												
TRPINT 015134	201	2518#											
TRPLIM 015246	2523	2573#											
TRPTAB 015176	2522	2532#											
TRPV 000034	201#												
TRP2 = 011000	738	2522	2530#										
TSECS 026106	3084#	3085#	3945#										
TSTMEN 030072	4182	4205#											
TSTM1 030200	4227#	4243											
TSTM2 030312	4254#	4271											
TSTM3 030400	4275#	4293											
TTPIV 000064	215#												
TTSRV1 007436	455	567	1456	1487#	1520	1521							
TTYBSY 001014	243#	562	729#	780#	983#	1038#	1108#						
TTYBYT 001260	346#	1345#	1346										
TTYLNK 001530	455#	1456#	1504#	1510#	1516#	1520#							
TTYSPV 001526	215	454#	3305										
TYPADR 001104	283#	1458#	1487	1500#									
TYPCTR 001110	285#	565#	1523#										
TYPDAT 004752	791	796	799	813	816	820	880	1008#					
TYPE 007366	772	779	805	950	954	956	967	971	974	1448	1456#	2715	2719
	3549	3562											
TYPEB 007476	1488	1500#											
TYPEC 007512	1503#												
TYPED 007562	1516#												
TYPEE 007556	1513	1515#	1519										
TYPEL 007610	1502	1505	1511	1517	1522#								
TYPEN 007350	763	927	936	944	1024	1033	1094	1101	1445#	3277	3293		
TYPEO 030072	147	728	1011	1260	1266	2491	2492	4018#					
TYPE1 007352	1446#	1449											
TYPLIM 031222	147#	726	1009	1264									
TYPLQ 001130	146#	147	2493	4018									



```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26

```

```

;
;
;
;
;
;
;
;
;
;
;
;
; IDENTIFICATION
;
; PRODUCT CODE: MAINDEC-11-DXQAE-D-LA
;
; PRODUCT NAME: DEC/X11 - SHORT MONITOR
;
; DATE: JANUARY 1977
;
; MAINTAINER: DIAGNOSTIC GROUP
;
;
;
;
; COPYRIGHT 1973, 1974, 1975, 1976, 1977 DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
;

```

```

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

```

```

;LIST SEQ
;ENABL AMA
;TITLE QAED DEC/X11 SHORT MONITOR
;ASRCT
;=0

;GLOBL LCR12,HCOR12,OACNV,BDCNV,MODQ,CLOCKL,CLOCKP
;GLOBL KW11L,KW11P,HMS,HOURS,WCASE,WCASEF,CLOCK
;GLOBL OFFSET,HUNGCK,HNGTIM,LCLEAR,PCLEAR,CHKSUM
;GLOBL TKS,TRB,TPS,TPB,FILCNT,FILLER,SR,RANNUM,CAST
;GLOBL MODCNT,MODCTR,PASREL,RELOC,ALLBK,KTPRES,PWRCNT

;*****IF RUNNING WITH AN 11/60 AND ANY SYSTEM TYPE ERROR OCCURS
; (TRAP TO 4, ETC.) "LOGBUF" WILL CONTAIN
; THE CONTENTS OF SCRATCHPAD LOCATIONS FROM 0-177. SCRATCHPAD
; LOCATIONS 100-107 ARE PRINTED OUT.

;SWITCH REGISTER OPTIONS:

;SR15=1 HALT MODULE AFTER ERROR.
;SR14=1 INHIBIT MODULE HALT AFTER 20 ERRORS.
;SR13=1 INHIBIT ERROR PRINT.
;SR12=1 ENABLE ENDPAS PRINTOUT.
;SR10=1 PRINT ALL DATA ERRORS IN BUFFER
;SR9=1 INHIBIT "RELOCATED TO " MESSAGE

;MODULE STATUS BIT DEFINITIONS:

; BIT 15=1 IOMOD
; BIT 14=1 MODULE SELECTED
; BIT 13=1 MODULE DROPPED
; BIT 12=1 EXTENDED MODULE
; BIT 11=1 DESELECT IF PROGRAM IS RELOCATED
; BIT 10=1 DESELECT IF NOT ON AN EVEN 32K OFFSET
; BIT 09=1 NRKMOD
; ALL 0'S SRKMOD

```

```

63          ;A FEW DEFINITIONS.
64
65          000000          R0   =#0   ;GENERAL PURPOSE REGISTER 0
66          000001          R1   =#1   ;GENERAL PURPOSE REGISTER 1
67          000002          R2   =#2   ;GENERAL PURPOSE REGISTER 2
68          000003          R3   =#3   ;GENERAL PURPOSE REGISTER 3
69          000004          R4   =#4   ;GENERAL PURPOSE REGISTER 4
70          000005          R5   =#5   ;GENERAL PURPOSE REGISTER 5
71          000006          R6   =#6   ;GENERAL PURPOSE REGISTER 6
72          000006          SP   =#6   ;POINTER TO HARDWARE (SYSTEM DEFAULT) STACK
73          000007          PC   =#7   ;PROGRAM COUNTER
74
75          177776          PS   =177776 ;PROCESSOR STATUS REGISTER
76          177776          PSW  =177776 ;PROCESSOR STATUS WORD
77          177570          HARDSR =177570 ;HARDWARE SWITCH REGISTER
78
79          177572          SSR0  =177572 ;MEMORY MANAGEMENT REGISTER 0
80          172516          SSR3  =172516 ;MEMORY MANAGEMENT REGISTER 3
81
82          170200          MAPREG =170200 ;BASE ADDRESS OF UNIBUS MAP BOX (11/70)
    
```

```

83          076600          MED   =76600 ;MAINTENANCE EXAMINE AND DEPOSIT INSTRUCTION (11/60)
84          000101          RDSERV =101  ;LOG SERVICE REG. ADR -- READ
85          000102          RDPBA  =102  ;LOG PHYSICAL BUSS ADR. REG. ADR. -- READ
86          000103          RDCUA  =103  ;LOG CUPRENT MICRO ADR. REG. ADR. -- READ
87          000104          RDFGIN =104  ;LOG FLAG/INTERRUPT REG. ADR. -- READ
88          000022          RDWHMI =022  ;GEN. WHO AM I REG. ADR. -- READ
89          000222          WTWHMI =222  ;GEN. WHO AM I REG. ADR. -- WRITE
90
91          177772          PIRORG =177772 ;PROGRAMMED INTERRUPT REQUEST REGISTER
92
93          140000          UMODE  =140000 ;BIT CONFIGURATION TO INDICATE USER MODE IN PSW
94          040000          SMODE  =40000  ;BIT CONFIGURATION TO INDICATE SUPERVISOR MODE IN PSW
95
96          172300          KIPDR0 =172300 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 0
97          172302          KIPDR1 =172302 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 1
98          172304          KIPDR2 =172304 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 2
99          172306          KIPDR3 =172306 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 3
100         172316          KIPDR7 =172316 ;KERNEL INSTRUCTION PAGE DESCRIPTOR REGISTER 7
101         172340          KIPAR0 =172340 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 0
102         172342          KIPAR1 =172342 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 1
103         172344          KIPAR2 =172344 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 2
104         172346          KIPAR3 =172346 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 3
105         172356          KIPAR7 =172356 ;KERNEL INSTRUCTION PAGE ADDRESS REGISTER 7
106
107         177600          UIPDR0 =177600 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 0
108         177602          UIPDR1 =177602 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 1
109         177604          UIPDR2 =177604 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 2
110         177616          UIPDR7 =177616 ;USER INSTRUCTION PAGE DESCRIPTOR REGISTER 7
111         177640          UIPAR0 =177640 ;USER INSTRUCTION PAGE ADDRESS REGISTER 0
112         177642          UIPAR1 =177642 ;USER INSTRUCTION PAGE ADDRESS REGISTER 1
113         177644          UIPAR2 =177644 ;USER INSTRUCTION PAGE ADDRESS REGISTER 2
114         177656          UIPAR7 =177656 ;USER INSTRUCTION PAGE ADDRESS REGISTER 7
    
```

115	000004	PIR08	=IOT	;MAKE THE PIR0 CALL A TRAP THROUGH THE IOT VECTOR
116	000000	OPEN	=0	;USE OPEN TO DECLARE ALL VARIABLES AS 0 BEFORE STARTING
117	100000	BIT15	=100000	;BIT 15 DEFINITION
118	040000	BIT14	=40000	;BIT 14 DEFINITION
119	020000	BIT13	=20000	;BIT 13 DEFINITION
120	010000	BIT12	=10000	;BIT 12 DEFINITION
121	004000	BIT11	=4000	;BIT 11 DEFINITION
122	002000	BIT10	=2000	;BIT 10 DEFINITION
123	001000	BIT9	=1000	;BIT 9 DEFINITION
124	000400	BIT8	=400	;BIT 8 DEFINITION
125	000200	BIT7	=200	;BIT 7 DEFINITION
126	000100	BIT6	=100	;BIT 6 DEFINITION
127	000040	BIT5	=40	;BIT 5 DEFINITION
128	000020	BIT4	=20	;BIT 4 DEFINITION
129	000010	BIT3	=10	;BIT 3 DEFINITION
130	000004	BIT2	=4	;BIT 2 DEFINITION
131	000002	BIT1	=2	;BIT 1 DEFINITION
132	000001	BIT0	=1	;BIT 0 DEFINITION
133				
134	000340	PRTY7	=340	;BIT CONFIGURATION FOR PRIORITY LEVEL 7 IN PSW
135	000300	PRTY6	=300	;BIT CONFIGURATION FOR PRIORITY LEVEL 6 IN PSW
136	000240	PRTY5	=240	;BIT CONFIGURATION FOR PRIORITY LEVEL 5 IN PSW
137	000200	PRTY4	=200	;BIT CONFIGURATION FOR PRIORITY LEVEL 4 IN PSW
138				
139	005746	PUSH	=005746	;TST -(SP) ;USED TO ADD A WORD TO THE STACK
140	024646	PUSH2	=024646	;CMP -(SP),-(SP) ;USED TO PUT TWO WORDS ON THE STACK
141	005726	POPSP	=005726	;TST (SP)+ ;USED TO DELETE A WORD FROM THE STACK
142	022626	POPSP2	=022626	;CMP (SP)+,(SP)+;USED TO DELETE TWO WORDS FROM THE STACK
143				
144	000100	IE	=BIT6	;DEFINITION OF INTERRUPT ENABLE BIT FOR ALL PDP-11 DEVICES
145	000040	KRUFLL	=32	;DEFINITION OF KEYBOARD BUFFER LENGTH (IN WORDS)
146	000310	IOQL	=200	;DEFINITION OF I/O SERVICE QUEUE LENGTH (IN BYTES)
147	000454	TYPQL	=300	;DEFINITION OF TYPING SERVICE QUEUE LENGTH (IN BYTES)
148	017572	TYPLIM	=TYPEQ+TYPQL	;DEFINITION OF LAST WORD IN TYPE QUEUE
149	017116	IOQLIM	=IOQ+IOQL	;DEFINITION OF LAST WORD IN I/O QUEUE

150				;THE FOLLOWING DEFINITIONS ARE THE INDEX VALUES USED BY MODE 6 AND 7
151				;MONITOR INSTRUCTIONS WHEN REFERRING TO LOCATIONS IN THE HEADER OF A MODULE
152				
153	000005	XFLAG	=5	;REFERENCE USED TO KEEP TRACK OF WRITE BUFFER USAGE
154	000020	STAT	=16	;POINTS TO THE LOW BYTE OF THE MODULE STATUS WORD
155	000021	STAT1	=17	;POINTS TO THE HIGH BYTE OF THE MODULE STATUS WORD
156	000022	INIT	=18	;POINTS TO THE MODULE'S STARTING ADDRESS
157	000024	SPOINT	=20	;POINTS TO THE MODULE'S STACK POINTER
158	000026	PSCNT	=22	;REFERENCES THE LOCATION KEEPING TRACK OF THE NUMBER OF PASSES
159	000030	ERCNT	=24	;REFERENCES THE LOCATION KEEPING TRACK OF THE NUMBER OF ERRORS
160	000032	SVR0	=26	;POINTS TO WHERE THE MODULE'S REGISTER 0 CONTENTS ARE STORED
161	000034	SVR1	=28	;POINTS TO WHERE THE MODULE'S REGISTER 1 CONTENTS ARE STORED
162	000036	SVR2	=30	;POINTS TO WHERE THE MODULE'S REGISTER 2 CONTENTS ARE STORED
163	000040	SVR3	=32	;POINTS TO WHERE THE MODULE'S REGISTER 3 CONTENTS ARE STORED
164	000042	SVR4	=34	;POINTS TO WHERE THE MODULE'S REGISTER 4 CONTENTS ARE STORED
165	000044	SVR5	=36	;POINTS TO WHERE THE MODULE'S REGISTER 5 CONTENTS ARE STORED
166	000046	SVR6	=38	;POINTS TO WHERE THE MODULE'S REGISTER 6 CONTENTS ARE STORED
167	000050	CSRA	=40	;REFERENCES THE BASE ADDRESS OF THE MODULE'S DEVICES(CONTROL AND
168	000052	ACSR	=42	;REFERENCES THE CONTENTS OF THE CONTROL AND STATUS REGISTER
169	000052	SRADR	=42	;POINTS TO THE ADDRESS OF WHAT THE DATA SHOULD HAVE BEEN
170	000054	ASTAT	=44	;POINTS TO ANYSTATUS REGISTER NOT THE SAME AS THE CSR
171	000054	WASADR	=44	;POINTS TO THE ADDRESS OF WHAT THE DATA WAS
172	000056	ASR	=46	;POINTS TO THE ACTUAL EXPECTED DATA
173	000060	AWS	=48	;POINTS TO THE ACTUAL RECEIVED DATA
174	000062	RSTR	=50	;POINTS TO THE MODULE RESTART ADDRESS
175	000064	RRUFVA	=52	;REFERENCES THE VIRTUAL ADDRESS OF THE READ BUFFER
176	000066	RBUFFA	=54	;POINTS THE LOCATION FOR THE PHYSICAL ADDRESS OF THE READ BUFFER
177	000070	RBUFFEA	=56	;POINTS THE EXTENDED ADDRESS BITS OF THE READ BUFFER
178	000072	RBUFFSZ	=58	;POINTS THE SIZE OF THE READ BUFFER
179	000074	WBUFFA	=60	;POINTS TO THE WRITE BUFFER PHYSICAL ADDRESS
180	000076	WBUFFEA	=62	;POINTS TO THE EXTENDED ADDRESS BITS OF THE WRITE BUFFER
181	000100	WRUFRO	=64	;REFERENCES THE AMOUNT OF WRITE BUFFER REQUESTED BY MODULE
182	000102	WBUFFSZ	=66	;REFERENCES THE SIZE OF THE WRITE BUFFER
183				
184				

```

185 ;LOAD TRAP AND VECTOR AREA
186
187 000000 000000 .#0
188 000000 000000 000000 .WORD 0,0
189 000004 012516 BUSEV: BUSERR ;BUS ERROR TRAP POINTER.
190 000006 000340 PRTY7
191 000010 012524 RESIV: RESINT ;RESERVED INSTRUCTION TRAP.
192 000012 000340 PRTY7
193 000014 004772 TRCV: TRCI ;TRACE TRAP POINTER.
194 000016 000000 0
195 000020 005510 IOTV: PIRQ. ;SERVICE PIRQ CALL
196 000022 000340 PRTY7 ;PREVENT LOWER LEVEL MODULE FROM COMING IN ON
197 ;TOP OF HIGHER LEVEL ONES
198 000024 012636 PWRFV: PWRDN ;POWER FAIL POINTER.
199 000026 000340 PRTY7
200 000030 000032 EMTV: .+2 ;EMT POINTER
201 000032 000000 HALT
202 000034 012404 TRPV: TRPINT ;TRAP POINTER.
203 000036 000000 0
204 000040 000040 .#40
205 000040 000000 DDPPTV: .WORD OPEN ;LOAD MEDIUM INDICATOR.
206 000042 000000 .WORD 0 ;CHAIN MODE ONLY, POINTS TO DDPHON.
207 000044 000000 .WORD 0
208 000046 012752 LOGIC
209 000050 000000 OPEN
210 000052 000000 OPEN
211 000054 000000G CAST ;GLOBAL POINTER FOR CAST MODIFICATION
212 000056 000001 1 ;ALSO FOR CAST
213 000060 .#60
214 000060 005644 KYBDV: KRSRVC ;KYBD INTERRUPT POINTER.
215 000062 000340 PRTY7
216 000064 001406 TTPIV: TTYSRV ;TELETYPE PRINTER SERVICE
217 000066 000200 PRTY4

```

```

218 ;FROM 70 THROUGH 776 FILLED WITH .+2 AND HALT.
219
220
221
222
223
224 ;LOAD STARTING ADDRESSES
225
226 000200 000137 001414 .#200
227 000200 000137 001414 JMP START ;GO TO START OF MONITOR.
228 001000 000137 001414 .#1000
229 001000 000137 001414 JMP START ;GO TO START OF MONITOR.
230
231
232 001004 000000 FAKESR: OPEN ;LOCATION OF PSEUDO SWITCH REG. WHEN USED

```

```

233 ;VARIABLE DATA AREA CLEARED EVERY TIME RUN IS ISSUED OR 'C' IS TYPED
234
235 001006 000 IOQUE: .RYTE OPEN ;IOQUE AND TYPQUE MUST BE IN SAME WORD!!
236 001007 000 TYPQUE: .RYTE OPEN
237 001010 000 SPCFLG: .RYTE OPEN ;SPCFLG AND DIRIND MUST BE IN SAME WORD!!
238 001011 000 DIRIND: .RYTE OPEN
239 001012 000 BKQUE: .RYTE OPEN
240 001013 000 BRAKE: .RYTE OPEN
241 001014 000 TTYBSY: .RYTE OPEN ;TELETYPE BUSY FLAG, 0= NOT BUSY.
242 001015 000 MODCNT: .RYTE OPEN
243 001016 000 MODCTR: .RYTE OPEN
244 001017 000 ERRIND: .RYTE OPEN ;0=ERROR, NOT 0=DATA ERROR.
245 001020 000 FILCTR: .RYTE OPEN
246 001021 000 MDXCTR: .RYTE OPEN ;EXTENDED I/O MODULES NOT YET RUN IN CURRENT BANK
247 001022 000 MDXCNT: .RYTE OPEN ;EXTENDED I/O MODULE COUNT
248 001023 000 WBFLLG: .RYTE OPEN ;CURRENT VALUE OF XFLAG BEING CHECKED FOR
249 001024 000 STOP: .RYTE OPEN ;STOP MODULES TO RELOCATE PROGRAM
250 001025 000 BKCNT: .RYTE OPEN ;BACKGROUND MODULE COUNT
251 001026 000 XCNT: .RYTE OPEN ;RUNNABLE EXTENDED MODULE COUNT (TO CALCULATE ROTCT)
252 001027 000 QUECTR: .RYTE OPEN
253 001030 000 XFRFLG: .RYTE OPEN
254 001031 000 ROTCT: .RYTE OPEN ;VALUE USED FOR BUFFER ROTATION TIMEOUT (16 TIMES
255 ;XCNT OR 177, WHICHEVER IS SMALLER)
256 001032 000 PASTOT: .RYTE OPEN ;ALTERNATE RELOCATION COUNTER- TOTAL ENDPASS
257 ;CALLS IN CURRENT BANK
258 001033 000 PASREL: .RYTE OPEN
259 001034 000 BKTMR: .RYTE OPEN ;BKMOD INSTR CTR- LOADED FROM BKTMR
260 001035 000 ROT: .RYTE OPEN ;WBUF ROTATION RATE CTR- LOADED FROM ROTN
261 .EVEN
262 001036 000000 KBPTR: OPEN
263 001040 000000 MODPTR: OPEN ;MODULE POINTER
264 001042 000000 ADDRLO: OPEN
265 001044 000000 ADDRHI: OPEN
266 001046 000000 NUMBER: OPEN
267 001050 000000 NUMBPA: OPEN
268 001052 000000 NUMBEA: OPEN
269 001054 000000 DSTADR: OPEN
270 001056 000000 SRETRN: OPEN
271 001060 000000 IOBKID: OPEN
272 001062 000000 YES: OPEN
273 001064 000000 TABADR: OPEN
274 001066 000000 RSTAT: OPEN
275 001070 000000 TRCPC: OPEN
276 001072 000000 TRCPSW: OPEN
277 001074 000000 IOQ1: OPEN ;I/O QUEUE POINTERS.
278 001076 000000 IOQ2: OPEN
279 001100 000000 TYPQ1: OPEN ;LOAD TYPE QUEUE POINTER
280 001102 000000 TYPQ2: OPEN ;UNLOAD TYPE QUEUE POINTER
281 001104 000000 TYPADR: OPEN
282 001106 000000 TYPRET: OPEN
283 001110 000000 TYPCTR: OPEN ;USED TO MONITOR CONSOLE TELETYPE TO PREVENT HANGING
284 001112 000000 OASCEA: OPEN ;USED TO HOLD EA BITS FOR OCTAL TO
285 ;ASCII CONVERSION
286 001114 000000 DERRCT: OPEN
287 001116 000000 SAVLO: OPEN
288 001120 000000 SAVHI: OPEN
    
```

```

289 001122 000000 MODADR: OPEN ;ADDRESS OF CURRENTLY EXECUTING (IOQ LEVEL) MODULE
290 001124 000000 TYPR1: OPEN ;SAVE LOCATIONS FOR TYPE ROUTINE REGISTERS
291 001126 000000 TYPR2: OPEN
292 001130 000000 TYPR3: OPEN
293 001132 000000 TYPR4: OPEN
294 001134 000000 TYPR5: OPEN
295 001136 000000 SPSAV: OPEN
296 001140 000000 ERRPOS: OPEN
297 001142 000000 CDPLG: OPEN ;INDICATE RETURN TO CHECKDATA ROUTINE
298 001144 000000 CCSRA: OPEN
299
300
    
```

```

301 ;VARIABLE AND FIXED DATA AREA NOT CLEARED WHEN RUN OR "C" IS ISSUED
302
303 001146 177560 TKS; 177560
304 001150 177562 TKB; 177562
305 001152 177564 TPS; 177564 ;IF THIS AND NEXT LOC ARE CHANGED,
306 001154 177566 TPB; 177566 ;CHANGE LOC,S LPWK1 AND LPWK2 IF PRESENT
307 001156 177570 SR; 177570 ;ADDRESS OF THE SWITCH REGISTER
308 001160 123456 RANNUM; 123456 ;RANDOM NUMBER IS CALC'ED AND PASSED HERE
309 001162 022333 RANWRK; 22333 ;ALSO USED BY RAND.
310 001164 000000 MEMVA; OPEN
311 001166 000000 MEMPA; OPEN
312 001170 000000 MEMEA; OPEN
313 001172 000000 MEM12; OPEN
314 001174 000000 MEMSBE; OPEN
315 001176 000000 MEMNAS; OPEN
316 001200 000000 LOCORE; OPEN ;STARTING ADDR OF FREE CORE (MULTIPLE OF 1000 OCTAL)
317 001202 000000 ZERO; 0 ;LOCORE EA BITS
318 001204 000000 LCOR12; OPEN ;TOP 12 BITS OF LOW CORE ADDRESS
319 001206 000000 HCOR12; OPEN ;CONTAINS ST. ADDR OF HIGHEST BUFFER,
; (TOP 12 BITS ONLY)
320
321 001210 000000 HCORSV; OPEN
322 001212 001414 WBUF; START ;CONTAINS CURRENT WRITE BUFFER ADDR.
323 001214 000000 WBFEA; OPEN ;WRITE BUFFER EA BITS (SHIFTED TO POS 4,5)
324 001216 000000 WBUF12; OPEN ;TOP 12 BITS OF WBUF ADDRESS IF USING ROTATION
325 001220 000000 OFFMX; OPEN ;TOP ALLOWABLE PROGRAM OFFSET
326 001222 000000 OFFSET; OPEN ;CURRENT PROGRAM OFFSET
327 001224 000000 OFFSTD; OPEN ;DESTINATION OFFSET
328 001226 000000 OFFSTM; OPEN ;SAVE LOCATION FOR DESTINATION OFFSET
; DURING MOVE
329
330 001230 000000 MOVANK; OPEN ;RANK BEING MOVED (SOURCE ADDRESS)
331 001232 000000 MAPRET; OPEN ;RETURN ADDRESS FROM MCODE ROUTINE
332 001234 000024 FERRIM; 20.
333 001236 006000 MXWBF; AMODNM-START/2&177000 ;MAX SIZE OF WRITE BUFFER, VARIES
; IF BUFFER ROTATION IS ENABLED
334
335 001240 000000 TTYBYT; OPEN
336 001242 000000 CXRET; OPEN ;CTLX RETURN ADDRESS
337 001244 000000 SYSCNT; OPEN ;TOTAL NUMBER OF SYSTEM ERRORS
338 001246 000000 PWRCNT; OPEN ;TOTAL NUMBER OF POWER FAILS
339 001250 000000 BKPTR; OPEN ;QUEUE POINTER INITIALIZATION FOR RUNNING BKMDS
340 001252 000000 KTPRES; OPEN ;0= INHIBITED OR NOT PRESENT (LEAVE AS FULL
; WORD FOR EASY MODIFICATION AND TO GUARANTEE
; THAT FILCNT AND FILLER ARE IN SAME WORD)
; FILCNT AND FILLER MUST BE IN SAME WORD
341
342
343 001254 014 FILCNT; .BYTE 12.
344 001255 000 FILLER; .BYTE 0
345 001256 000 SYSERI; .BYTE OPEN
346 001257 000 RMODE; .BYTE OPEN ;RUN MODE INDICATOR
347 001260 000 CHN; .BYTE OPEN
348 001261 000 ROTL; .BYTE OPEN
349 001262 000 ROTL; .BYTE OPEN ;WRITE BUFFER ROTATION LOCKED FLAG
350 001263 000 RELOC; .BYTE OPEN ;RELOCATION ALLOWED FLAG
351 001264 000000 ROTCNT; OPEN ;ROTATION COUNTER USED WITH ROTL
352 001266 000100 ROTNUM; 100 ;ROTATION CONSTANT USED WITH ROTCNT
353 001270 000400 ROTSI2; 400 ;ROTATE PROGRAM BY RK
354 001272 000 FILLID; .BYTE OPEN ;FILL INDICATOR.
355 001273 000 MOVING; .BYTE OPEN ;CODE BEING RELOCATED
356 001274 000 RUNRFL; .BYTE OPEN

```

```

357 001275 000 LOCK; .BYTE OPEN ;PREVENT ROTATION OF CODE EXECUTION
358 001276 000 MVDOWN; .BYTE OPEN
359 001277 000 RETRY; .BYTE OPEN ;RETRY COUNT FOR CODE RELOCATION
360 001300 010 QUECNT; .BYTE 10 ;# OF TIMES TO CHECK QUEUE BEFORE CHECKING
; TYPEQ WHEN BOTH HAVE REQUESTS PENDING
361
362 001301 000 WDL45; .BYTE OPEN ;NON-ZERO IF CPU IS 11/45
363 001302 000 GWRFC; .BYTE OPEN ;WRITE BUFFER ROTATION COUNT
364 001303 000 MEMERP; .BYTE OPEN
365 001304 010 BKTIME; .BYTE 10
366 001305 003 ROTN; .BYTE 3 ;# OF GWRFC CALLS (LESS ONE) NEEDED PER
; ROTBUF CALL
367
368 001306 000 PPRES; .BYTE OPEN ;PARITY PRESENT, 0=INHIBITED OR NONEXISTENT
369 001307 000 CNT1; .BYTE OPEN
370 001310 000 ATE; .BYTE OPEN
371 001311 000 WFLAG; .BYTE OPEN
372 001312 000 HNGFLG; .BYTE OPEN ;FLAG INDICATES CKHUNG ROUTINE IS BUSY
373 001313 000 MOVFLG; .BYTE OPEN ;FLAG USED IN MCODE ROUTINE
374 001314 000 MSGFLG; .BYTE OPEN ;FLAG INDICATING MSGN OP MSGS CALL
375 001315 000 DATFLG; .BYTE OPEN ;FLAG INDICATING CKDATA OR DACKCALL
376 001316 012 SYSLIM; .BYTE 10. ;MAX. NUMBER OF SYSTEM ERRORS ALLOWED
377 001317 000 TFLAG; .BYTE OPEN ;TIMER FLAG INDICATING ERROR OR EOP
378 001320 000 NSTOP; .BYTE OPEN ;FLAG PREVENTING STOP FROM BEING SET
379 001321 001 CLOCK; .BYTE 1 ;CLOCK FLAG SET FOR SHORT MONITOR
380 001322 000 ALLBK; .BYTE OPEN ;WHEN NON 0 MEANS RUN ALL BKMDS BEFORE RELOCATING
381
382 001324 001324 CLOCKL; KW11L ;WILL BE 0 IF NO KW11-L CLOCK
383 001326 000000G CLOCKP; KW11P ;WILL BE 0 IF NO KW11-P CLOCK
384
385 001330 001604 HNGTIM; 900. ;HUNG TIME LIMIT IN SECONDS
386 001332 000000G ALCLR; LCLEAR ;SUBROUTINE ADDRESS IN L CLOCK MODULE
387 001334 000000G APCLR; PCLEAR ;SUBROUTINE ADDR IN P CLOCK MODULE
388 ;WILL DROP ANY I/O MODULE INACTIVE FOR 15. MINS.

```

```

389 ; *** DO NOT CHANGE THE SEQUENCE OF THE NEXT 9 WORDS *** ;
390 ;
391 ;REFER TO THE DOCUMENT FOR THE MONITOR CHECKSUM MODULE (BKBA)
392 ;FOR THE USE OF THE NEXT SIX WORDS, AND HOW THE TWO "OPEN" LOCATIONS
393 ;ARE DEFINED WHEN A NEW VERSION OF THE MONITOR IS RELEASED
394
395 001336 137577          CHK12: 137577          ;LOW CHKSUM FOR PROC. WITH NO RTT INST.
396 001340 002641          CHK2: 2641           ;HIGH CHKSUM
397 001342 001414          CHKSUM: .WORD START      ;BEGIN ADDR OF MONITOR PURE AREA
398 001344 015232          .WORD WCASEE           ;END ADDR OF MONITOR PURE AREA
399 001346 137607          CHK11: 137607         ;CONTAINS LOW CKSM OF MONITOR PURE AREA (PROC WITH RTT)
400 001350 002641          CHK1: 2641           ;CONTAINS HI CKSM OF MONITOR PURE AREA
401 .ODD
402 001353 000003          HOURS: .BLKB 3          ;HOURS IN ASCII
403 001356 000000          MINITS: OPEN           ;MINUTES IN ASCII
404 001360 000000          SEKNS: OPEN           ;SECONDS IN ASCII
405
406 .EVEN
  
```

```

407 ;VARIABLE QUE CALL AREA- KEPT IN FRONT OF STAPT SINCE IT IS IMPURE CODE
408 ;
409 ;COMMON "QUE CALL" ROUTINE, DOES A DIRECT DISPATCH RATHER THAN
410 ;REQUEING DUE TO MULTIPLE POINTS CALLING THIS ROUTINE WHICH
411 ;MAY OVERLAP IF REQUEUED.
412
413 001362 012700 001372    COMQUE: MOV      #CADDR,PO
414 001366 000137 002500    CADDR: JMP      IOQSVN
415 001372 000000          CADDR: OPEN           ;DESTINATION ADDR.
416 001374 000000          CSTART: OPEN          ;MODULE START ADDR. (0 FOR MONITOR).
417
418 ;CHECKDATA ERROR CALL
419 001376 104421          CDERR: DERR16
420 001400 000000          CDERR1: OPEN          ;ADDRESS OF CKDATA CALL
421 001402 000137 007202    .JMP CDPET
422
423 ;TELETYPE ROUTINE PIRO CALL
424 ;TTYLNK IS FREQUENTLY MODIFIED DURING EXECUTION
425
426 001406 000004          TTYSRV: PIRO6
427 001410 076426          TTYLNK: TTSRV1
428 001412 000000          0
  
```

```

429 ;STARTUP SEQUENCE
430
431 001414 000005          START:  RESET          ;CLEAR THE WORLD.
432 001416 012706 016546  MOV      #SPBOT,R6    ;SET UP STACK.
433 001422 012737 012636 000024  MOV      #PWRDN,PWRFV  ;SET UP POWER FAIL VECTOR.
434 001430 012746 000004  MOV      #BUSEV,=(SP)  ;SAVE PRESENT BUS ERR TRAP
435 001434 012737 177570 001156  MOV      #HARDSR,SR    ;SET SR=HARDWARE SR
436 001442 012737 001456 000004  MOV      #3,BUSEV     ;SET NEW BUS ERR TRAP
437 001450 005777 177502  TST     #SR           ;IS IT HARDWARE SR?
438 001454 000404  BR      4#           ;IF YES, RESTORE TRAP
439 001456 012737 001004 001156 3# :  MOV      #FAKESR,SR   ;ELSE SET UP SOFTWARE SR
440 001464 022626  CMP      (SP)+,(SP)+  ;RESTORE STACK
441 001466 012637 000004 4# :  MOV      (SP)+,BUSEV  ;RESTORE BUS ERR TRAP
442 001472 012737 016234 001250  MOV      #MODQ=2,BKPTR ;INITIALIZE TO START BKMODS WITH 1ST ONE
443 001500 005737 001200  TST     LDCORE       ;DONE BUFFER SETUP?
444 001504 001002  BNE     2#           ;BR IF YES.
445 001506 004537 016606  JSR     R5,SETBUF   ;NO, DO IT.
446 001512 004737 012272 2# :  JSR     PC,CLRQUS   ;CLEAR QUEUES.
447 001516 105037 001257  CLR     RMODE
448 001522 112737 177777 001021  MOV     #-1,MDCXTR  ;INITIALIZE COUNTER
449 001530 105037 001262  CLR     ROTL        ;MAKE SURE ROTATION IS NOT LOCKED
450 001534 104406 014062  MSG#,TITLE          ;TYPE THE TITLE
451 001540 022737 001004 001156  CMP     #FAKESR,SR  ;DO WE HAVE A REAL SR?
452 001546 001002  BNE     55#         ;BR IF YES
453 001550 104406 014756  MSG#,NOSWR         ;ELSE TELL HIM WHERE IT IS
454 001554 004737 013262 55# :  JSR     PC,ROTCHK   ;CHECK IF ROTATF ALLOWED AND TYPE RANGE.
455
456 001560 105737 000041  TST     ##41       ;ANY LOAD MEDIUM RITS SET ?
457 001564 001402  BEQ     1#           ;NO, CONTINUE
458 001566 104406 014431  MSG#,CLR40         ;" TO EXERCISE LOAD MEDIUM YOU MUST CLEAR LOC 40
459 001572 005037 001224 1# :  CLR     OFFSTD     ;MAKE SURE PROGRAM IS NOT PARTIALLY RELOCATED
460 001576 004737 014060  JSR     PC,MVCODE   ;CLEAR QUEUES IN CASE MVCODE OVER WROTE THEM
461 001602 004737 012272  JSR     PC,CLRQUS   ;INITIALIZE VECTOR AREA
462 001606 004737 012354  JSR     PC,CLRVEC   ;KT11 IN USE ?
463 001612 105737 001252  TST     KTPRES      ;YES, CONTINUE
464 001616 001002  BNE     5#           ;NO, LOAD WORST CASE PATTERN IN BUFFER
465 001620 004737 015016  JSR     PC,LOADWC   ;TYPE DOT.
466 001624 104406 014122 5# :  MSG#,DOT          ;CHAIN MODE?
467 001630 105737 001260  TST     CHN         ;NO- GO AWAIT KEYBOARD INPUT
468 001634 001405  BEQ     INPUT       ;CLEAN OUT KEYBOARD BUFFER WITH A CR
469 001636 112777 000015 177172 6# :  MOV     #15,RKBPTR  ;YES-START RUNNING.
470 001644 000137 007420  JMP     RUN
471
472 001650 012706 016546  INPUT:  MOV      #SPROT,R6 ;RESET STACK.
473 001654 105037 001256  CLR     SYSERR      ;CLEAR SYSTEM ERROR INDICATOR.
474 001660 012737 016306 001036  MOV      #KRUF,KRPTR ;INITIALIZE KYBD BUFFER POINTER.
475 001666 052777 000100 177252  RIS      #IE,#TKS   ;ENABLE KYBD INTERRUPTS.
476 001674 113737 001300 001027  MOV     QUECNT,QUECTR
477
    
```

```

478 ;QUEUE CHECK. NOTE THAT IF IOBKID IS NEGATIVE, NO TYPE QUE REQUESTS WILL BE SERVICED
479
480 001702 105337 001027  QUETST:  DECB     QUECTR ;COUNT TO PREVENT BREAK CALLS FROM LOCKING OUT TYPING
481 001706 003004  BGT     1#           ;ALSO PREVENTS HIGH SPEED INT DEVICES FROM
482 ;LOCKING OUT TYPE QUEUE AND R/HN SERVICE
483 ;STARTUP OF MODULES
484
485 001710 113737 001300 001027  MOV     QUECNT,QUECTR ;RESET QUE COUNTER
486 001716 000405  BR      2#           ;
487 001720 105737 001006 1# :  TST     IOQUE       ;IO QUE REQUEST PENDING?
488 001724 001402  BEQ     2#           ;
489 001726 000137 002424 2# :  JMP     IOQSVC     ;GO SERVICE IF YES.
490 001732 005737 001060  TST     IOBKID      ;STARTING I/O MODULES?
491 001736 100422  BMI     QTSTC       ;BR IF YES.
492 001740 105737 001007 3# :  TST     TYPQUE     ;TYPE REQUEST PENDING?
493 001744 001412  BEQ     QTSTB       ;BR IF NOT.
494 001746 105737 001014  TSTR   TTYBSY      ;TTY BUSY?
495 001752 001002  BNE     QTSTA       ;BR IF YES.
496 001754 000137 002650  JMP     TYP SVC     ;NO, GO SERVICE TYPF QUEUEF.
497 001760 005337 001110  QTSTA:  DEC     TYPCTR ;CHECK FOR TTY HUNG
498 001764 001002  BNE     QTSTB       ;
499 001766 000137 006426  JMP     TTSRV1     ;UNHANG TTY
500 001772 005737 001070  QTSTB:  TST     TRPC  ;BACKGROUND MODULE PENDING?
501 001776 001402  BEQ     QTSTC       ;
502 002000 000137 002622  JMP     BKQ SVC     ;JUMP IF YES
503 002004 105737 001257  QTSTC:  TSTR   RMODE  ;IN RUN MODE?
504 002010 001734  BEQ     QUETST     ;BR IF NOT, GO CHECK AGAIN.
    
```

```

505 ;RUN MODE SERVICE ROUTINE,
506 ;NOTE THAT MODULE'S STACK POINTER IS REINITIALIZED BUT OTHER
507 ;REGISTERS HAVE ANY PREVIOUS VALUES RESTORED AT START OR RESTART
508
509 002012 105737 001013 RUNSVCI TSTB BRAKE ;IS THE BRAKE ON?
510 002016 001331 BNE QUETST ;BR IF YES, DO NOT INIT MORE MODULES,
511 002020 105737 001322 TSTB ALLBK ;RUN ALL BKMODS BEFORE RELOCATING ?
512 002024 001407 BEQ 12# ;NO, CONTINUE
513 002026 105737 001024 TSTB STOP ;WAITING TO RELOCATE ?
514 002032 001407 BEQ 13# ;NO, CONTINUE
515 002034 012737 040020 001060 MOV #40020,IOBKID ;START UP ONLY BKMOD'S
516 002042 000415 BR 1# ;CONTINUE
517
518 002044 105737 001024 12# TSTR STOP ;STOP SET?
519 002050 001314 BNE QUETST ;YES, DON'T START ANY MORE MODULES
520 002052 105737 001016 13# TSTB MODCTR ;MODCTR #0?
521 002056 001007 BNE 1# ;BR IF NOT,
522 002060 105737 001260 TSTB CHN ;YES, ARE WE IN CHAIN MODE?
523 002064 100004 BPL 1# ;BR IF NOT,
524 002066 012737 040020 001060 MOV #40020,IOBKID ;SWITCH IT TO BACKGROUND AND PREVENT LOOPING
525 002074 000702 BR QUETST ;THRU BACKGROUND MODULE SERIES MORE THAN ONCE
526 002076 062737 000002 ADD #2,MODPTR ;POINT TO NEXT MODULE.
527 002104 017700 176730 MOV #MODPTR,RO ;MODULE ADDR TO RO.
528 002110 001474 REQ 7# ;BR IF NO ADDR.
529 002112 016046 MOV STAT(RO),-(SP) ;GET MODULE'S STATUS
530 002116 042716 016000 RIC #RIT12|RIT11|BIT10,(SP) ;MASK SPECIAL INDICATORS
531 002122 022637 001060 CMP (SP)+,IOBKID ;CORRECT MODULE TYPE TO RUN?
532 002126 001134 BNE 11# ;NO= BRANCH
533 002130 005737 001222 TST OFFSET ;IF OFFSET 0, IGNORE BITS 10 +11
534 002134 001414 BEQ 2# ;CONTINUE
  
```

```

535 002136 032760 004000 000020 BIT #RIT11,STAT(RO) ;RIT11 SET?
536 002144 001125 RBE 11# ;YES= DON'T RUN MODULE SINCE OFFSET NOT 0
537 002146 032737 001777 001222 BIT #1777,OFFSET ;OFFSET 32K MULTIPLE?
538
539 002154 001404 REQ 2# ;YES= IGNORE BIT 10
540 002156 032760 002000 000020 BIT #RIT10,STAT(RO) ;NO= BIT10 SET?
541 002164 001115 BNE 11# ;YES= DON'T RUN MODULE
542 002166 005737 001060 2# TST IOBKID ;BACKGROUND MODULE?
543
544
545
546 002172 100406 BMI 3# ;BR IF NOT.
547 002174 112737 177777 001013 MOVB #-1,BRAKE ;YES, APPLY BRAKE.
548 002202 013737 001040 001250 MOV MODPTR,BKPTR ;RECORD MOD0 ADDRESS OF CURRENT BKMOD
549 002210 032760 010000 000020 3# BIT #RIT12,STAT(RO) ;EXTENDED I/O MODULE?
550 002216 001402 BEQ 4# ;NO= BRANCH
551 002220 105737 001022 INCB MDXCNT ;YES, COUNT IT
552 002224 105060 000005 4# CLRFB XFLAG(RO) ;CLEAR INDICATOR
553 002230 016060 000024 000046 MOV SPOINT(RO),SVR6(RO) ;SET UP MODULE SP POINTER,
554 002236 105737 001274 TSTB RUNPFL
555 002242 001004 BNE 5#
556 002244 016037 000022 001054 44# MOV INIT(RO),DSTADR ;SET UP DESTINATION ADDR.
557 002252 000407 BR 6#
558 002254 016037 000062 001054 5# MOV RSTRT(RO),DSTADR
559 002262 005760 000026 TST PSCNT(RO) ;HAS MODULE BEEN THRU START YET?
560 002266 001001 BNE 6# ;BR IF YES
561 002270 000765 BR 44# ;ELSE GO BACK
562 002272 105337 001016 6# DECB MODCTR ;DECR COUNT OF MODS INITED.
563 002276 000137 002514 JMP IOQSVE ;GO TO START MODULE
564 002302 022737 040000 001060 7# CMP #40000,IOBKID ;FINISHED WITH SPECIAL BKMOD ?
565 002310 001004 BNE 8# ;BR IF NOT.
566 002312 012737 041000 001060 MOV #41000,IOBKID ;YES, SWITCH TO NBKMOD
567 002320 000434 BR 10# ;CONTINUE
568 002322 022737 041000 001060 8# CMP #41000,IOBKID ;FINISHED WITH NBKMODS ?
569 002330 001013 BNE 9# ;NO, CONTINUE
570 002332 012737 140000 001060 MOV #140000,IOBKID ;YES, SWITCH TO IOMOD.
571 002340 112737 177777 001021 MOVB #-1,MDXCTR ;INITIALLY LOCK MDXCTR
572
573
574
575
576 002346 105037 001022 CLRB MDXCNT ;INITIALIZE COUNT
577
578
579
580
581
582 002352 105037 001320 CLRB NSTOP ;ALLOW STOP FLAG TO BE SET
583 002356 000415 BR 10# ;CONTINUE
584 002360 005737 001060 9# TST IOBKID ;IF ALREADY BACKGROUND
585 002364 100012 BPL 10# ;DON'T RESET MDXCTR
586
587
588
589
590 002366 113737 001022 001021 MOVB MDXCNT,MDXCTR ;SETUP MDXCTR
  
```

```

591 002374 012737 040020 001060      MOV      #40020,IOBKID ;SWITCH TO BACKGROUND MODE
592
593
594
595
596 002402 013737 001250 001040      MOV      BKPTR,MODPTR ;CONTINUE WITH NEXT BKMOD
597          MOV      #1,RUNRFL ;SET RUN RESTART FLAG11/60 *****
598 002410 000403          BR       11$
599 002412 012737 016234 001040 108$  MOV      #MOD0=2,MODPTR ;POINT TO MODULE TABLE START.
600 002420 000137 001702 118$  JMP      QUETST
    
```

```

601          ;I/O QUE SERVICE ROUTINE.
602          ;SERVICES PIRQs,QUEs, AND BREAKS CALLS
603          ;NOTE THAT PIRQ SERVICE CAN OCCUR ON TOP OF THIS ROUTINE- SO
604          ;LEVEL 7 PRIORITY IS REQUIRED IF COUNT IS DECREMENTED AFTER IOQ2
605          ;IS MOVED
606
607 002424 023727 001076 017116 IOQSV0: CMP      IOQ2,#IOQLIM ;REACHED LIMIT OF QUEUE?
608 002432 103403          BLO     1$ ;RR IF NOT.
609 002434 012737 016606 001076      MOV      #IO0,IOQ2 ;RESET IOQ2.
610 002442 125337 001006 18$  DECB   IOQUE ;DECREMENT REQUEST COUNT
611 002446 017700 176424          MOV      #IOQ2,R0 ;GET PC+2 OF CALL.
612 002452 062737 000002 001076      ADD     #2,IOQ2 ;UPDATE IOQ2.
613 002460 010001          MOV      R0,R1 ;SAVE PC+2 OF CALL
614 002462 014046          MOV      -(R0),-(SP) ;PUSH CALL ON STACK
615 002464 022726 174407          CMP     #BREAK$, (SP)+ ;BREAK CALL?
616 002470 001002          BNE     2$ ;NO, CONTINUE
617 002472 000137 003136          JMP     BREAK, ;YES, GO SERVICE IT
618 002476 005720 28$  TST     (R0)+
619
620 002500 012037 001054          IOQSV1: MOV      (R0)+,DSTADR ;GET DESTINATION ADDR.
621 002504 012000          MOV      (R0)+,R0 ;GET MODULE ADDR. IS IT 0?
622 002506 001902          BNE     IOQSV2 ;RR IF NOT. IT'S A MODULE.
623 002510 000177 176340          JMP     #DSTADR ;GO DO MONITOR FUNCTION.
624 002514 116037 000020 001066 IOQSV2: MOV      STAT(R0),RSTAT ;GET RUN STATUS.
625 002522 032760 020000 000020 IOQSV3: BIT     #BIT13,STAT(R0) ;MODULE STOPPED?
626 002530 001402          BEQ     1$ ;NO, CONTINUE
627 002532 000137 001702          JMP     QUETST ;YES, FORGET IT
628 002536 032760 040000 000020 18$  BIT     #BIT14,STAT(R0) ;IS MODULE SELECTED ?
629 002544 001002          BNE     2$ ;YES, CONTINUE
630 002546 000137 001702          JMP     QUETST ;NO, FORGET IT
631 002552 010037 001122 28$  MOV      R0,MODADR ;SAVE R0. (MODULE ADDR).
632 002556 010637 001136          MOV      R6,SPSAV ;SAVE MONITOR STACK TOO!.
633 002562 062700 000050          ADD     #SVR6+2,R0 ;RESTORE MODULE'S REGS.
634 002566 014006          MOV      -(R0),R6 ;STARTING WITH STACK POINTER.
635 002570 014005          MOV      -(R0),R5
636 002572 014004          MOV      -(R0),R4
637 002574 014003          MOV      -(R0),R3
638 002576 014002          MOV      -(R0),R2
639 002600 014001          MOV      -(R0),R1
640 002602 014000          MOV      -(R0),R0
641 002604 013746 001066 IOQSV4: MOV      RSTAT,-(SP) ;LOAD RUN STATUS.
642 002610 105066 000001          CLR    1(SP) ;KERNEL, 1ST REGISTER SET
643 002614 013746 001054          MOV      DSTADR,-(SP) ;LOAD DESTINATION ADDR.
644 002620 000006          RTTI   RTI ;GO TO DESTINATION.
    
```

```

        ,BACKGROUND QUEUE SERVICED HERE.
    645
    646
    647 002622 013737 001070 001054 BKQ8VC: MOV TRCPD,DSTADR ;SET UP DESTINATION ADDR.
    648 002630 113737 001072 001066          MOVB TRCPDW,RSTAT ;SET UP RUN STATUS,
    649 002636 017700 176176          MOV  #MODPTR,R0 ;MODULE START ADDR TO R0.
    650 002642 005037 001070          CLR TRCPD ;CLEAR BK MODULE WAITING INDICATOR.
    651 ;IF ERROR OCCURS, WILL STOP MODULE FROM FURTHER
    652 ;EXECUTION UNTIL ERROR IS REPORTED
    653 002646 000725          BR IOOSVA ;GO GET GOING.
    
```

```

        ;TYPE QUE SERVICE ROUTINE.
        ;REMOVES PC+2 OF CALL FROM Q, PASSING IT IN R1
    654
    655
    656
    657 002650 023727 001102 017572 TYP8VC: CMP TYPQ2,#TYPLIM ;REACHED UPPER END OF QUEUE?
    658 002656 103403          BLO IS ;RR IF NOT.
    659 002660 012737 017116 001102          MOV #TYPEQ,TYPQ2 ;YES, RESET TYPQ2.
    660 002666 105237 001014          INCB TTYBSY ;INDICATE TTY BUSY.
    661 002672 017701 176204          MOV #TYPQ2,R1 ;GET PC+2 OF CALL.
    662 002676 062737 000002 001102          ADD #2,TYPQ2 ;UPDATE TYPQ2.
    663 002704 105337 001007          DECB TYPQ2 ;DECREMENT REQUEST COUNT.
    664 002710 010146          MOV R1,-(SP) ;PUSH PC+2 OF CALL
    665 002712 005741          TST -(R1) ;POINT TO CALL.
    666 002714 010137 001046          MOV R1,NUMBER ;SAVE ADDRESS OF CALL
    667 002720 011101          MOVW (R1),R1 ;GET CALL
    668 002722 006301          ASL R1 ;TIMES 2.
    669 002724 016101 171726          MOV TYPTAB-TRP2-4(R1),R1 ;FORM SERVICE ADDR.
    670 002730 000201          RTS R1 ;GO TO IT, RESTORE R1.
    671
    672 002732 004626          TYPTAB: .WORD PASEND
    673 002734 004440 003524 003516          .WORD ENDSVC,ERR8VC,ERSVC1
    674 002742 003110 000001 003506          .WORD MSG,,1,ERSVC2,MSGN,,1,1,1,1,MSG,,1,1,DERR,,MSGD.
    675 002750 003004 000001 000001
    676 002756 000001 000001 002774
    677 002764 000001 000001 003244
    678 002772 003150
    
```

```

;MSGN. ROUTINE. SERVICES CALLS TO TYPE ASCII MESSAGES.
679
680
681 002774 112737 177777 001314 MSGS.1  MOV  #-1,MSGFLG  ;SET MSGS CALL FLAG
682 003002 000402          BR  MSGCONT  ;CONTINUE
683 003004 105037 001314 MSGN.1  CLRB  MSGPLG  ;SET MSGN CALL FLAG
684
685 003010 012137 001064 MSGCONT: MOV  (R1)+,TABADR  ;GET ASCII TABLE ADDR.
686 003014 012137 001374          MOV  (R1)+,CSTART  ;MODULE ADDR TO CSTART.
687 003020 010137 001372          MOV  R1,CADDR  ;RESUME ADDR TO CADDR.
688 003024 013701 001374          MOV  CSTART,R1  ;MODULE ADDR TO R1.
689 003030 032777 020000 176120 BIT  #BIT13,0SR  ;INHIBIT ERROR PRINT?
690 003036 001033          BNE  MSG1.  ;BR IF YES.
691 003040 105737 001314          TSTB MSGPLG  ;IS THIS A MSGS CALL ?
692 003044 100410          BMI  1$  ;YES, CONTINUE
693 003046 004737 004314          JSR  PC,ENDCOM ;NO, DO COMMON STUFF.
694 003052 004537 006340          JSR  RS,TYPEN  ;TYPE
695 003056 014127          CRCP          ;? CARRAGE RETURNS
696 003060 015541          AEND          ;COMMON HEADER.
697 003062 015576          $PASS        ;PASS #.
698 003064 000000          0
699 003066 013701 001064 10:  MOV  TABADR,R1  ;TABLE ADDR TO R1.
700 003072 012146          20:  MOV  (R1)+,(SP) ;GET MESSAGE ADDR.
701 003074 022716 177777          CMP  #-1,(SP)  ;TERMINATOR?
702 003100 001412          BEQ  MSG1.  ;BR IF YES, DONE.
703 003102 004737 006356          JSR  PC,TYPE  ;NO. TYPE MESSAGE.
704 003106 000771          BR  2$  ;GO DO IT AGAIN.
    
```

```

;MSG# CALL SERVICE# HERE
705
706
707 003110 012146          MSG.1  MOV  (R1)+,(SP)  ;ASCII MESSAGE ADDR TO STACK.
708 003112 010137 001372          MOV  R1,CADDR  ;RESUME ADDR TO CADOR.
709 003116 005037 001374          CLR  CSTART  ;INDICATE MONITOR QUE CALLING.
710 003122 004737 006356          JSR  PC,TYPE  ;GO TYPE DESIRED MESSAGE.
711 003126 105037 001014 MSG1.1  CLRB  TTYBSY
712 003132 000137 001362          JMP  COMQUE  ;GO QUEUE UP TO RESUME.
    
```

```

713 ;BREAK ROUTINE. SERVICES BREAK CALL.
714
715 003136 012100 BREAK, MOV (R1)+,R0 ;GET MODULE ADDR.
716 003140 010137 001054 MOV R1,DSTADR ;GET DESTINATION ADDR.
717 003144 000137 002514 JMP IOGSVE ;QUE UP TO RESUME
  
```

```

718 ;MSGD& SERVICE= TYPE DATA ERROR COUNT AND TRANSFER SIZE
719 ;AND HALT MODULE IF NECESSARY
720
721 003150 010137 001372 MSGD, MOV R1,CADDR ;SET UP RESUME ADDRESS
722 003154 004737 004376 JSP PC,TYPDAT ;GET MODULE ADDRESS
723 003160 012601 MOV (SP)+,R1
724 003162 010137 001374 MOV R1,CSTART ;SAVE ADDRESS
725 003166 004537 011206 JSR R5,FILLNM ;GET MODULE NAME
726 003172 016141 AERNM+1
727 003174 004737 004376 JSR PC,TYPDAT ;SETUP ERROR COUNT (ASCII)
728 003200 004537 011744 JSR R5,RDCNV
729 003204 016153 AERCT
730 003206 004737 004376 JSR PC,TYPDAT ;SETUP TOTAL WORDS TRANSFERRED
731 003212 004537 011744 JSR P5,RDCNV
732 003216 016205 AERTOT
733 003220 032777 020000 175730 BIT #R1I3,@SR ;INHIBIT ERROR PRINTOUT?
734 003226 001004 RNE MSGD1, ;YES= BRANCH
735 003230 012746 016140 MOV #AERNM,-(SP) ;NO= TYPE MESSAGE
736 003234 004737 006356 JSR PC,TYPE
737 003240 000137 004222 MSGD1, JMP ERSVND ;CHECK FOR HALTING MODULE, AND RETURN IF NOT
  
```

```

738 ;DERRX, ROUTINE, SERVICES EXTENDED DATA ERROR CALLS FROM THE MONITOR
739 ;CHECKDATA ROUTINE, TYPES ERROR INFORMATION AND RETURNS
740
741 003244 112737 000001 001017 DERRX:1 MOVB #1,ERRIND ;POINT TO RETURN LOCATION
742 003252 005721 TST (R1)+ ;SETUP RESUME ADDRESS
743 003254 010137 001372 MOV R1,CADDR ;GET MODULE ADDRESS
744 003260 004737 004376 JSR PC,TYPDAT ;AND SAVE IT
745 003264 012601 MOV (SP)+,R1 ;SETUP ADDRESS OF CKDATA CALL
746 003266 010137 001374 MOV R1,CSTART ;AND SAVE IT
747 003272 004737 004376 JSR PC,TYPDAT ;SETUP ADDRESS OF CKDATA CALL
748 003276 012637 001046 MOV (SP)+,NUMBER ;SETUP ADDRESS OF CKDATA CALL
749 003302 004737 004314 JSR PC,ENDCOM
750 003306 012702 000005 MOV #5,R2
751 003312 004737 004376 DERRX:1 JSR PC,TYPDAT
752 003316 005302 DEC R2
753 003320 001374 BNE DERRX1
754 003322 016146 000030 MOV ERCNT(R1),-(SP) ;ERROR COUNT TO STACK
755 003326 004537 011744 JSR R5,BDCNV ;CONVERT ERROR COUNT TO DECIMAL
756 003332 016052 AERNMB
757 003334 005046 CLR =(SP)
758 003336 005766 TST 2(SP)
759 003342 001402 BEQ 18
760 003344 012716 000060 MOV #60,(SP) ;CSRA EA BITS
761 003350 004537 011524 18: JSR R5,OACNVX ;CSR ADDRESS= ASCII
762 003354 015722 ADTE6
763 003356 012603 MOV (SP)+,R3 ;SAVE TABLE ADDRESS
764 003360 012637 001140 MOV (SP)+,ERRPOS ;SAVE OFFSET COUNT
765 003364 006337 001140 ASL ERRPOS
766 003370 012346 MOV (R3)+,-(SP) ;SETUP READ BUFFER PA
767 003372 012346 MOV (R3)+,-(SP) ;GET EA BITS
768 003374 063766 001140 000002 ADD ERRPOS,2(SP)
769 003402 103002 BCC 28
770 003404 062716 000020 ADD #20,(SP)
771 003410 005723 TST (R3)+ ;SKIP SIZE
772 003412 004537 011524 28: JSR R5,OACNVX ;GET PA OF RDRADR
773 003416 016002 ADTE4
774 003420 012346 MOV (3)+,-(SP) ;GET WRITE PA
775 003422 012346 MOV (3)+,-(SP) ;GET EA BITS
776 003424 063766 001140 000002 ADD ERRPOS,2(SP)
777 003432 103002 BCC 38
778 003434 062716 000020 ADD #20,(SP)
779 003440 004537 011524 38: JSR R5,OACNVX ;GET ASCII OF WRADR
780 003444 015765 ADTE5
781 003446 004537 011512 JSR R5,OACNV ;CONVERT S/B TO ASCII
782 003452 015735 ADTE3
783 003454 004537 011512 JSR R5,OACNV ;CONVERT WAS TO ASCII
784 003460 015750 ADTE2
785 003462 013746 001140 MOV ERPOS,-(SP) ;ERROR POSITION IN BUFFER
786 003466 000241 CLR ;CORRECT BACK FROM BYTE TO WORD PTR
787 003470 006216 ASR (SP)
788 003472 005216 INC (SP) ;MAKE 1ST WORD #1
789 003474 004537 011744 JSR R5,BDCNV ;GET DECIMAL
790 003500 016034 ADTE7
791 003502 000137 004014 JMP ERSVX ;GO TO COMMON ROUTINE

```

```

792 ;ERROR CALLS ARE SERVICED HERE.
793 ;R1 AT ENTRY CONTAINS PC+2 OF CALL
794
795 003506 112737 000003 001017 ERVVC2:1 MOVB #3,ERRIND ;INDICATE ERROR CALL.
796 003514 000406 BR ERVVC1 ;INDICATE DATA ERROR.
797 003516 105037 001017 ERVVC1: CLRR EPRIND
798 003522 000403 BR ERVVC1
799 003524 112737 000002 001017 ERVVC2:1 MOVB #2,ERRIND ;INDICATE "NORMAL" ERROR.
800 003532 012137 001374 ERVVC1:1 MOV (R1)+,CSTART ;SAVE START ADDR OF MODULE.
801 003536 122737 000003 001017 ERVVC2:1 CWPB #3,ERRIND ;RFORM CALL?
802 003544 001002 BNE 18 ;BP IF NOT
803 003546 012137 001084 MOV (R1)+,TABADR ;SAVE TABLE ADDR.
804 003552 010137 001372 MOV R1,CADDR ;RESUME ADDR TO PSEUDO.
805 003556 013701 001374 MOV CSTART,R1 ;GET BACK START ADDR.
806 003562 004737 004314 JSR PC,ENDCOM ;DO COMMON STUFF.
807 003566 005261 000030 INC ERCNT(R1) ;INCREMENT MODULE'S ERROR COUNT.
808 003572 001002 BNE ERVVCB ;BP IF RESULT NOT 0.
809 003574 005161 000030 COM ERCNT(R1) ;RESET COUNT TO -1.
810 003600 012702 000005 ERVVCB:1 MOV #5,R2 ;GET TYPE DATA FROM QUEUE TO STACK.
811 003604 004737 004376 ERVVC1:1 JSR PC,TYPDAT ;DO IT.
812 003610 005302 DEC R2 ;DONE?
813 003612 001374 BNE ERVVC ;BP IF NOT.
814 003614 016146 000030 MOV ERCNT(R1),-(SP) ;ERROR COUNT TO STACK.
815 003620 004537 011744 JSR R5,BDCNV ;CONVERT ERROR COUNT TO DECIMAL.
816 003624 016052 AERNMB
817 003626 105737 001017 TSTB ERRIND ;DATA ERROR?
818 003632 001047 BNE 28 ;BP IF NOT.
819 003634 004537 011512 JSR R5,OACNV ;CONVERT WAS TO ASCII.
820 003640 015750 ADTE2
821 003642 004537 011512 JSR R5,OACNV ;CONVERT S/B TO ASCII,RESET TYPQ2.
822 003646 015735 ADTE3
823 003650 012637 001046 MOV (SP)+,NUMBER ;GET ADDR OF LOCATION READ
824 003654 104413 GETPAS,NUMBER
825 003660 013746 001050 MOV NUMBPA,-(SP)
826 003664 013746 001052 MOV NUMBEA,-(SP)
827 003670 004537 011524 JSR R5,OACNVX ;CONVERT WASADR TO ASCII.
828 003674 016002 ADTE4
829 003676 012637 001046 MOV (SP)+,NUMBER
830 003702 104413 GETPAS,NUMBER
831 003706 013746 001050 MOV NUMBPA,-(SP)
832 003712 013746 001052 MOV NUMBEA,-(SP)
833 003716 004537 011524 JSR R5,OACNVX ;CONVERT SPADR TO ASCII.
834 003722 015765 ADTE5
835 003724 005046 CLR =(SP) ;0 ONTO STACK
836 003726 016637 000002 001144 MOV 2(SP),CCSRA ;SAVE CSR ADDRESS
837 003734 001402 BEQ 18 ;IF 0, BRANCH
838 003736 012716 000060 MOV #60,(SP) ;SETUP EA BITS
839 003742 004537 011524 18: JSR R5,OACNVX ;CONVERT CSR ADDR TO ASCII.
840 003746 015722 ADTE6
841 003750 000421 BR ERSVX
842 003752 022671 28: MOV (SP)+,-(SP)+ ;SKIP WAS AND S/B.
843 003754 004537 011512 JSR R5,OACNV ;CONVERT STAT REG CONTENTS TO ASCII.
844 003760 011746 ADTE7
845 003762 004537 011512 JSR R5,OACNV ;CONVERT CSR CONTENTS TO ASCII.
846 003766 015687 ACSRC
847 003770 005046 CLR =(SP) ;0 ONTO STACK

```

848	003772	016637	000002	001144		MOV	2(SP),CCSRA	SAVE CSR ADDRESS
849	004000	001402				SEG	38	BRANCH IF ADDRESS 0
850	004002	012716	000060			MOV	#60,(SP)	
851	004006	004537	011524		38:	JSR	R5,OACNVX	CONVERT CSR ADDR TO ASCII.
852	004012	015653					ACSRAC	
853	004014	032777	020000	175134	ERSVX:	BIT	#BIT13,0SR	INHIBIT ERROR PRINT?
854	004022	001077				BNE	EPSVND	BR IF YES.
855	004024	004537	006340		78:	JSR	R5,TYPE	TYPE
856	004030	014127				CRCR		2 CARRAGE RETURNS
857	004032	015541				AEND		COMMON HEADER.
858	004034	015576				#PASS		PASS #.
859	004036	016044				ERRNMB		ERROR #.
860	004040	000000				0		
861	004042	122737	000002	001017	88:	CMPB	#2,ERRIND	DATA ERROR?
862	004050	003406				BLE	18	BR IF NOT.
863	004052	004537	006340			JSR	R5,TYPE	YES, TYPE
864	004056	016012				ADTE4A		DATA ERROR VALUES
865	004060	015714				ADTERR		
866	004062	000000				0		
867	004064	000404				BR	28	
868	004066	012746	015645		18:	MOV	#ERROR,-(SP)	TYPE ERROR MESSAGE.
869	004072	004737	006356			JSR	PC,TYPE	
870	004076	122737	000001	001017	28:	CMPB	#1,ERRIND	EXTENDED DATA ERROR?
871	004104	001004				BNE	38	NO- BRANCH
872	004106	012746	016026			MOV	#ADTEX,-(SP)	YES- OUTPUT POSITION IN BUFFER
873	004112	004737	006356			JSR	PC,TYPE	
874	004116	012746	014155		38:	MOV	#ACRLF,-(SP)	
875	004122	004737	006356			JSR	PC,TYPE	
876	004126	122737	000003	001017		CMPB	#3,ERRIND	ERROR CALL?
877	004134	001032				BNE	ERSVND	BR IF NOT.
878	004136	013702	001064			MOV	TABADR,R2	TABLE ADDR TO R2.
879	004142	012703	000010		48:	MOV	#R,R3	WILL TYPE B VALUES PER LINE.
880	004146	022712	177777		58:	CMP	#1,(2)	TERMINATOR?
881	004152	001417				BEO	68	BR IF YES, DONE.
882	004154	013246				MOV	R(2)+,-(SP)	PUT VALUE IN STACK.
883	004156	004537	011512			JSR	R5,OACNV	CONVERT IT TO OCTAL.
884	004162	015515				HOLD6		AND PUT IT HERE
885	004164	012746	015515			MOV	#HOLD6,-(SP)	TYPE IT.
886	004170	004737	006356			JSR	PC,TYPE	
887	004174	005303				DEC	R3	DONE B PER LINE?
888	004176	001363				BNE	58	BR IF NOT.
889	004200	012746	014155			MOV	#ACRLF,-(SP)	OUTPUT CRLF.
890	004204	004737	006356			JSR	PC,TYPE	
891	004210	000754				BP	48	GO FOR MORE.
892	004212	012746	014155		68:	MOV	#ACRLF,-(SP)	OUTPUT CRLF.
893	004216	004737	006356			JSR	PC,TYPE	
894	004222	005777	174730		ERSVND:	TST	#SR	HALT MODULE ON ERROR?
895	004226	100410				BMI	18	BR IF YES.
896	004230	023761	001234	000030		CMP	EPRLIM,EPCNT(R1)	PROP COUNT 20 OR GREATER?
897	004236	003010				BGT	28	BR IF NOT, CONTINUE MODULE EXECUTION.
898	004240	032777	040000	174710		BIT	#BIT14,#SR	YES, HALT MODULE AFTER 20 ERRORS?
899	004246	001004				BNE	28	NO- SKIP
900	004250	005037	001144		18:	CLP	CCSRA	CLEAR CCSRA
901	004254	000137	004446			JMP	ENDSVA	YES, GO HALT MODULE
902	004260	105037	001014		28:	CLRR	TTYBSY	QUE TO RESUME
903	004264	005737	001144			TST	CCSRA	IF NOT SETUP, DON'T RESTORE INTO MODULE

904	004270	001407				REQ	38	
905	004272	013700	001374			MOV	CSTART,R0	
906	004276	013760	001144	000050		MOV	CCSRA,CSRA(P0)	RESTORE CSRA IN MODULE
907	004304	005037	001144			CLP	CCSRA	
908	004310	000137	001362		38:	JMP	CONQUE	

909	004314	104413	001046	ENDCON:	GETPA\$,NUMBER		;NUMBER CONTAINS ADDRESS OF CALL
910	004320	013746	001050		MOV	NUMBPA,-(SP)	
911	004324	013746	001052		MOV	NUMBEA,-(SP)	
912	004330	013746	001046		MOV	NUMBER,-(SP)	
913	004334	160116			SUB	R1,(SP)	;COMPUTE ASSEMBLY PC.
914	004336	004537	011512		JSR	RS,OACNV	;CONVERT ASSEMBLY PC TO ASCII.
915	004342	015566			AEND2		
916	004344	004537	011524		JSR	RS,OACNVX	;CONVFRT PC TO ASCII.
917	004350	015553			AEND1		
918	004352	004537	011206		JSR	RS,FILLNM	;LET'S GET MODULE NAME.
919	004356	015541			AEND		;STUFF AT AEND.
920	004360	016146	000026		MOV	PSCNT(R1),-(SP)	;GET PASS COUNT.
921	004364	005216			INC	(SP)	;UP IT 1.
922	004366	004537	011744		JSR	RS,BDCNV	;CONVERT IT TO DECIMAL ASCII.
923	004372	015606			APASS		
924	004374	000207			RTS	PC	;LET'S GET OUT.

925							;TYPDAT ROUTINE LOADS QUEUED DATA ONTO STACK.
926							
927	004376	011646		TYPDAT:	MOV	(SP),-(SP)	;SAVE EXIT ON STACK AGAIN.
928	004400	023727	001102	017572	CMF	TYPQ2,#TYPLIM	;REACHED END OF QUEUE?
929	004406	103403			BLO	16	;BR IF NOT.
930	004410	012737	017116	001102	MOV	#TYPEQ,TYPQ2	;YES, POINT TO START OF QUEUE.
931	004416	017766	174460	000002	18:	MOV	#TYPQ2,2(SP)
932	004424	062737	000002	001102	ADD	#2,TYPQ2	;UPDATE QUEUE POINTER.
933	004432	105337	001007		DECB	TYPQ2	;DECREMENT COUNT.
934	004436	000207			RTS	PC	;EXIT, LEAVE DATA IN STACK.

```

935 ;END CALL SERVICED HERE.
936
937 004440 011101 ENDSVC: MOV (R1),R1 ;GET START ADDR.
938 004442 004737 JSR PC,ENDCOM ;DO COMMON STUFF.
939 004446 052761 020000 000020 ENDSVA: BIS #BIT13,STAT(R1) ;SET STOP BIT IN MODULE STAT.
940
941 004454 004537 006340 18: JSR R5,TYPEN ;TYPE
942 004460 014127 CRCP ;2 CARRAGE RETURNS
943 004462 015541 AEND ;COMMON HEADER.
944 004464 015615 MODEND ;END MESSAGE.
945 004466 000000 0
946 004470 105037 001014 28: CLRB TTYBSY ;CLEAR TTY BUSY INDICATOR.
947 004474 005761 000020 ENDSVE: TST STAT(R1) ;BACKGROUND MODULE?
948 004500 100410 BMI 18 ;BR IF NOT.
949 004502 105037 001013 CLRBR BRAKE ;RELEASE BRAKE.
950 004506 105737 001024 TSTB STOP ;STOP MODULES FOR RELOCATION?
951 004512 001403 REG 18 ;NO, CONTINUE
952 004514 105737 001022 TSTB ALLBK ;RUN ALL BKMODS BEFORE RELOCATION ?
953 004520 001413 REG ENDSVG ;NO, CONTINUE
954 004522 032761 010000 000020 18: BIT #BIT12,STAT(R1) ;EXTENDED I/O MOD?
955 004530 001402 BEQ 68 ;NO- BRANCH
956 004532 105337 001022 DECB MDXCNT ;YES, DEC EXTENDED MOD COUNT
957 004536 105337 001015 MODCNT ;DECR COUNT OF MODULES RUNNING.
958 004542 105737 001024 TSTB STOP ;RELOCATE CODE?
959 004546 001428 BEQ ENDSVF ;NO- BRANCH
960 004550 123737 001015 001025 ENDSVG: CMPB MDXCNT,BKCNT ;ALL I/O MODULES OFF?
961 004556 003021 BGT ENDSVD ;NO- BRANCH
962 004560 105737 001322 TSTB ALLBK ;RUN ALL BKMOD'S BEFORE RELOCATING ?
963 004564 001404 REG 18 ;NO, CONTINUE
964 004566 105737 001015 TSTB MDXCNT ;ALL MODULES DONE ?
965 004572 001404 BEQ ENDSVB ;YES, GO RELOCATE
966 004574 000412 BR ENDSVD ;NO, GO RUN THE REST OF THE MODULES
967 004576 105737 001013 18: TSTB BRAKE ;YES- BKMOD OFF?
968 004602 001007 BNE ENDSVD ;NO- BRANCH (STOP BKMOD CLEANLY)
969 004604
970 004604 000137 007476 ENDSVB: JMP RUNRES ;GO STARTUP AGAIN
971 004610 105737 001015 ENDSVF: TSTB MDXCNT ;ANY MODULES RUNNING?
972 004614 001002 BNE ENDSVD ;YES- BRANCH
973 004616 000137 006142 JMP CTRLCB ;COUNT 0, TERMINATE RUN MODE.
974 004622 000137 001702 ENDSVD: JMP QUITST ;GO BACK TO SERVICE QUEUES.
    
```

```

975 ;PASEND ROUTINE. TYPES END OF PASS MESSAGE.
976 ;BACKGROUND MODULES ARE NOT ALLOWED TO MAKE MULTIPLE PASSES.
977 ;IN CHAIN MODE NO MODULE ALLOWED TO MAKE MULTIPLE PASSES.
978
979 004626 011101 PASEND: MOV (R1),R1 ;LOAD R1 WITH MODULE ADDRESS
980 004630 016137 000062 001372 MOV RSTRT(R1),CADDR ;SAVE RESTART ADDRESS
981 004636 010137 001374 MOV R1,CSTART ;SAVE MODULE ADDRESS
982 004642 005261 000026 174302 18: INC PSCNT(R1) ;INCREMENT PASS COUNT.
983 004646 032777 010000 BIT #BIT12,#BR ;ENDPAS PRINTOUT?
984 004654 001417 BEQ PSENDA ;BR IF NOT.
985 004656 004737 004314 JSR PC,ENDCOM ;DO COMMON STUFF.
986 004662 016146 000026 MOV PSCNT(R1),-(SP) ;NOW GET IT.
987 004666 004537 011744 JSR R5,BDCNV ;CONVERT IT TO DECIMAL ASCII.
988 004672 015636 BPSCNT ;STUFF IT AT BPSCNT.
989 004674 004537 006340 JSR R5,TYPEN ;TYPE
990 004700 014127 CRCP ;2 CARRAGE RETURNS
991 004702 015541 AEND ;COMMON HEADER.
992 004704 015627 PSEND ;ENDPAS AND COUNT.
993 004706 014125 CR
994 004710 000000 0
995 004712 000400 BR PSENDA ;CONTINUE
996 004714 105037 001014 PSENDA: CLRR TTYBSY ;CLEAR TTY BUSY INDICATOR.
997 004720 105737 001260 CHN ;IN CHAIN MODE?
998 004724 100663 RMI ENDSVE ;BR IF YES TO END MODULE EXECUTION
999 004726 005761 000020 28: TST STAT(R1) ;BACKGROUND MODULE?
1000 004732 100415 BMI PSEND ;BR IF NOT.
1001 004734 105037 001013 CLRBR BRAKE ;RELEASE BRAKE.
1002 004740 032761 001000 000020 BIT #BIT9,STAT(P1) ;IS IT AN BKMOD ?
1003 004746 001725 BEQ ENDSVD ;NO, CONTINUE
1004 004750 052761 020000 000020 BIS #BIT13,STAT(R1) ;YES, SET IT'S STOP BIT
1005 004756 105337 001025 DECB BKCNT ;ONE LESS BKMOD RUNNING
1006 004762 000137 004474 JMP ENDSVE ;GO TAKE CARE OF OTHER STUFF
1007 004766 000137 001362 PSEND: JMP COMQUE ;GO TO COMMON QUE CALL.
1008
1009
    
```

```

1010                                     ;TRACE TRAP ENTERS HERE.
1011
1012 004772 105337 001034      TRCI:  DECB   BKTIMP
1013 004776 100401              TRCIA:  BMI    TRCIC
1014 005000 000006              TRCIA:  RTI
1015 005002 113737 001304      TRCIC:  MOVB   BKTIME,BKTIMER
1016 005010 005737 001006      TRCIC:  TST    IOQUE
1017 005014 001771              TRCIB:  BEQ    TRCIA
1018 005016 012637 001070      TRCIB:  MOV    (SP)+,TRCPC
1019 005022 012637 001072      TRCIB:  MOV    (SP)+,TRCPSW
1020 005026 000401              TRCIB:  BR     EXIT1,
    ;I/O OR TYPE QUE WAITING?
    ;NO. EXIT.
    ;SAVE MOD'S PC.
    ;SAVE MOD'S PSW.
    
```

```

1021                                     ;EXIT CALL ENTERS HERE.
1022
1023 005030 022626      EXIT:  CMP    (SP)+,(SP)+
1024 005032 010046      EXIT1: MOV    R0,-(SP)
1025 005034 013700 001122      EXIT1: MOV    MODADR,R0
1026 005040 062700 000032      EXIT1: ADD    #SVRO,R0
1027 005044 012620      EXIT1: MOV    (SP)+,(R0)+
1028 005046 010120      EXIT1: MOV    R1,(R0)+
1029 005050 010220      EXIT1: MOV    R2,(R0)+
1030 005052 010320      EXIT1: MOV    R3,(R0)+
1031 005054 010420      EXIT1: MOV    R4,(R0)+
1032 005056 010520      EXIT1: MOV    R5,(R0)+
1033 005060 010610      EXIT1: MOV    R6,(R0)
1034 005062 013706 001136      EXIT1: MOV    SP$AV,R6
1035 005066 005046      EXIT2: CLR    -(SP)
1036 005070 012746 005076      EXIT2: MOV    #16,-(SP)
1037 005074 000002      EXIT2: RTI
1038 005076 000137 001702      EXIT2: JMP    QUETST
    ;SAVE R0 IN STACK.
    ;MODULE ADDR TO R0.
    ;POINT TO MOD'S REG SAVE AREA.
    ;SAVE R0. (FROM STACK).
    ;SAVE REMAINING REGS.
    ;SAVE MODULE STACK POINTER.
    ;RESTORE MONITOR STACK.
    ;CLEAR PSW ON STACK
    ;SET UP RETURN PC
    ;CLEAR PSW AND CONTINUE
    
```

```

1039                                     ;TYPQ4. ROUTINE. LOADS MSGD& INFORMATION INTO TYPEQ
1040
1041 005102 004737 005350 TYPQ4.i JSR PC,LDTYPQ ;QUEUE PC+2 OF CALL
1042 005106 013700 001122 MOV MODADR,R0 ;LOAD MODULE ADDRESS IN R0
1043 005112 010046 MOV R0,-(SP) ;AND SAVE IN QUEUE
1044 005114 004737 005350 JSR PC,LDTYPQ
1045 005120 016046 000060 MOV ANAS(R0),-(SP) ;QUEUE DATA ERROR COUNT
1046 005124 004737 005350 JSR PC,LDTYPQ
1047 005130 016046 000056 MOV ASB(R0),-(SP) ;QUEUE SIZE OF TRANSFER
1048 005134 004737 005350 JSR PC,LDTYPQ
1049 005140 005726 POPSP ;REMOVE PS FROM STACK
1050 005142 013706 001136 MOV SPSAV,R6 ;GET MONITOR STACK POINTER
1051 005146 000747 BR EXIT2. ;EXIT

```

```

1052                                     ;TYPQ3. ROUTINE QUEUES INFORMATION FOR DERRX CALLS
1053
1054 005150 011646 TYPQ3.i MOV (SP),-(SP)
1055 005152 004737 005350 JSR PC,LDTYPQ ;QUEUE PC+2 OF CALL
1056 005156 013700 001122 MOV MODADR,R0 ;LOAD MODULE ADDRESS IN R0
1057 005162 010046 MOV R0,-(SP) ;AND SAVE IN QUEUE
1058 005164 004737 005350 JSR PC,LDTYPQ
1059 005170 017616 000000 MOV R(R0),(SP) ;QUEUE VIRTUAL ADDRESS OF CHECKDATA CALL
1060 005174 004737 005350 JSR PC,LDTYPQ
1061 005200 005726 POPSP
1062 005202 062700 000062 ADD #AWAS+2,R0 ;GET ADDRESS OF INFORMATION
1063 005206 012701 000005 MOV #S,R1
1064 005212 000401 BR 2$
1065 005214 005010 1$: CLR (R0) ;CLEAR DATA (EXCEPT LAST)
1066 005216 014046 2$: MOV =(R0),-(SP) ;GET DATA
1067 005220 004737 005350 JSR PC,LDTYPQ ;QUEUE DATA
1068 005224 005301 DFC R1
1069 005226 001372 BNE 1$
1070 005230 013700 MOV MODADR,R0
1071 005234 010660 000046 MOV R6,SVR6(R0)
1072 005240 013706 001136 MOV SPSAV,R6
1073 005244 000137 005066 JMP EXIT2.

```

1074  
1075  
1076 005250 011646  
1077 005252 004737 005350  
1078 005256 010046  
1079 005260 017600 000002  
1080 005264 062700 000050  
1081 005270 012746 000005  
1082 005274 012046  
1083 005276 000402  
1084 005300 011046  
1085 005302 005020  
1086 005304 004737 005350  
1087 005310 005316  
1088 005312 001372  
1089 005314 005726  
1090 005316 012600  
1091 005320 000137 005030

;TYPQ2, ROUTINE SERVICES ERROR, DATA ERROR, AND ERRN CALLS.  
TYPQ2: MOV (SP),=(SP) ;SAVE PC+2 OF CALL AGAIN,  
JSR PC,LDTYPQ ;QUEUE UP PC+2 OF CALL,  
MOV R0,=(SP) ;SAVE R0.  
MOV #2(SP),R0 ;GET MODULE ADDR,  
TYPQX: ADD #CSRA,R0 ;  
MOV #5,=(SP) ;  
MOV (R0)+,=(SP)  
BP 28  
18: MOV (R0),=(SP)  
CLR (R0)+  
28: JSR PC,LDTYPQ  
DEC (SP)  
BNE 18  
POPSP  
MOV (SP)+,R0 ;RESTORE R0.  
JMP EXIT.

1092  
1093  
1094 005324 004737 005350  
1095 005330 005726  
1096 005332 000137 005032

;TYPQ, ROUTINE SERVICES FND, ENDPAS, AND \*SG\* CALLS.  
TYPQ: JSR PC,LDTYPQ ;QUEUE UP CALL.  
POPSP  
JMP EXIT.

```

1097 ;TYPQ1. ROUTINE, SERVICES MSG CALL.
1098
1099 005336 004737 005350 TYPQ1, JSR PC,LDIYPO ;QUEUE UP CALL.
1100 005342 005726 POPSP
1101 005344 000137 005066 JMP EXIT2.
  
```

```

1102 ;ROUTINE TO LOAD TYPE QUEUE. ENTERED VIA JSR PC.
1103 005350 005046 LDYPO: CLR -(SP) ;CLEAR PSM ON STACK
1104 005352 012746 005360 MOV #38,-(SP) ;SETUP RETURN PC
1105 005356 000002 RTI ;CLEAR PSM AND CONTINUE
1106 005360 105737 001007 38: TSTB TYPQUE ;REQUEST COUNT 0?
1107 005364 001414 BEQ 1$ ;BR IF YES.
1108 005366 023737 001100 001102 CMP TYPQ1,TYPQ2 ;NO, TYPQ1 AND TYPQ2 SAME?
1109 005374 001010 BNE 1$ ;BR IF NOT.
1110 005376 012737 017116 001100 MOV #TYPEQ,TYPQ1 ;RESET THE TYPE QUEUE
1111 005404 104406 014157 MSGS,QOVRFL ;YES, QUEUE OFLO, CRASH SYSTEM
1112 005410 104406 014620 MSGS,HLT ;'HALT'
1113 005414 000000 HALT
1114 005416 023727 001100 017572 1$: CMP TYPQ1,#TYPLIM ;REACHED HIGH LIMIT?
1115 005424 001003 BNE 2$ ;BR IF NOT.
1116 005426 012737 017116 001100 MOV #TYPEQ,TYPQ1 ;RESET TYPQ1.
1117 005434 016677 000002 173436 2$: MOV 2(SP),#TYPQ1 ;STORE ARGUMENT IN QUEUE
1118 005442 105237 001007 INCB TYPQUE ;UPDATE REQUEST COUNTS.
1119 005446 062737 000002 001100 ADD #2,TYPQ1 ;UPDATE TYPQ1.
1120 005454 012616 MOV (SP)+,(SP) ;
1121 005456 000207 RTS PC ;EXIT.
  
```

```

1122 ;ROUTINE TO LOAD BREAK CALL
1123
1124 005460 112766 177777 000003 LDBRK, MOVB #1,3(SP) ;INDICATE BREAK CALL
1125 005466 000402 BR LDIOQ
1126
1127 ;ROUTINE TO LOAD QUE CALL.
1128
1129 005470 005066 000002 QUE, CLR 2(SP) ;INDICATE QUE CALL.
1130
1131 ;LDIOQ ROUTINE. MUST NOT DROP BELOW LEVEL OF INTERRUPTING DEVICE.
1132
1133 005474 012746 000340 LDIOQ: MOV #PRTY7,-(SP) ;PUT NEW PSW ON STACK
1134 005500 012746 005906 MOV #16,-(SP) ;SETUP RETURN PC
1135 005504 000002 RTI ;LOAD NEW PSW AND CONTINUE
1136 005506 000403 BR
1137 005510 112766 000001 000002 PIRQ, MOVB #1,2(SP)
1138 005516 105737 001006 TSTB IOQUE ;REQUEST COUNT 0?
1139 005522 001414 BEQ 10 ;BR IF YES.
1140 005524 023737 001074 001076 CMP IOQ1,IOQ2 ;IOQ1 AND IOQ2 SAME?
1141 005532 001010 BNE 10 ;BR IF NOT.
1142 005534 012737 016606 001074 MOV #IOQ,IOQ1 ;RESET IOQ1
1143 005542 104406 014157 MSGS,QOVRFL ;QUE OFLO, CRASH SYSTEM
1144 005546 104406 014620 MSGS,HLT ;'HALT'
1145 005552 000000 HALT
1146 005554 023727 001074 017116 10: CMP IOQ1,#IOQLIM ;REACHED HIGH LIMIT?
1147 005562 001003 BNE 20 ;BR IF NOT.
1148 005564 012737 016606 001074 MOV #IOQ,IOQ1 ;RESET IOQ1.
1149 005572 012677 173276 20: MOV (SP)+,#IOQ1 ;STORE PC+2 OF PENDING CALL.
1150 005576 105237 001006 INCR IOQUE ;UPDATE REQUEST COUNTS.
1151 005602 062737 000002 001074 ADD #2,IOQ1 ;UPDATE IOQ1.
1152 005610 005716 TST (SP) ;QUE CALL?
1153 005612 001003 BNE 30 ;NO- BRANCH
1154 005614 005726 TST (SP)+ ;YES, POP STACK
1155 005616 000137 005066 JMP EXIT2. ;EXIT
1156 005622 126627 000001 177777 30: CMPB 1(SP),#-1 ;BREAK CALL?
1157 005630 001003 BNE 40 ;NC- BRANCH
1158 005632 005726 TST (SP)+ ;YES- POP STACK
1159 005634 000137 005032 JMP EXIT1.
1160 005640 005726 40: TST (SP)+ ;PIRQ- EXIT VIA RTI
1161 005642 000002 RTI
    
```

```

1162 ;KEYBOARD INTERRUPT SERVICE
1163
1164 005644 005077 173276 KBSRVC: CLR #TKS ;CLEAR INTERRUPT ENABLE.
1165 005650 017746 173274 MOV #TKB,-(SP) ;GET CHAR.
1166 005654 042716 000200 BIC #200,(SP) ;REMOVE EXTRA BITS.
1167 005660 122716 000003 CMPB #3,(SP) ;IS IT CTL C?
1168 005664 001524 BEQ CTRLCA ;BR IF YES.
1169 005666 005726 POPSP ;REMOVE CHAR FROM STACK.
1170 005670 000004 005676 000000 PIRQ0,10,0 ;Q DEVICE SERVICE.
1171
1172 ;KEYBOARD SERVICE.
1173
1174 005676 023727 001036 016346 10: CMP KBPTR,#KBUF+KBUFL ;BUFFER EXCEEDED?
1175 005704 001511 BEQ 120 ;BR IF YES.
1176 005706 017746 173236 MOV #TKB,-(SP) ;GET CHAR.
1177 005712 042716 000200 BIC #200,(SP) ;CLEAR PARITY BIT.
1178 005716 121627 000177 CMPB (SP),#177 ;RUBOUT?
1179 005722 001443 BEQ 50 ;BR IF YES.
1180 005724 121627 000040 CMPB (SP),#40 ;SPACE? (IGNORE LEADING ONES).
1181 005730 001460 BEQ 80 ;BR IF YES.
1182 005732 121627 000015 CMPB (SP),#15 ;CR? (TIME TO QUIT?).
1183 005736 001462 BEQ 90 ;BR IF YES.
1184 005740 121627 000012 CMPB (SP),#12 ;LF? (TREAT LIKE CR).
1185 005744 001457 BEQ 90 ;BR IF YES.
1186 005746 121627 000060 CMPB (SP),#60 ;CHAR LESS THAN 60?
1187 005752 103425 BLO 100 ;BR IF YES. IGNORE IT.
1188 005754 121627 000141 CMPB (SP),#141 ;CHAR LESS THAN LOWER CASE A?
1189 005760 103405 BLO 20 ;BR IF YES.
1190 005762 121627 000172 CMPB (SP),#172 ;CHAR HIGHER THAN LOWER CASE Z?
1191 005766 101017 BHI 100 ;BR IF YES.
1192 005770 162716 000040 SUB #40,(SP) ;LOWER CASE CHAR. MAKE IT UPPER CASE.
1193 005774 111677 173036 20: MOVB (SP),#KBPTR ;STORE CHAR.
1194 006000 005237 001036 INC KBPTR ;UPDATE BUFFER POINTER.
1195 006004 112637 001240 MOVB (SP)+,TTYBYT ;CHAR TO ECHO BUFFER.
1196 006010 104406 001240 MSGS,TTYBYT ;ECHO THE CHARACTER.
1197 006014 052777 000100 173124 40: BIS #IE,#TKS ;REENABLE KYBD INTERRUPTS.
1198 006022 000137 001702 JMP QUETST
1199 006026 005726 100: POPSP
1200 006030 000771 BR 40
1201
1202 ;SERVICE DELETE/RUBOUT CODE.
1203
1204 006032 022737 016306 001036 50: CMP #KBUF,KBPTR ;BUFFER EMPTY?
1205 006040 001405 BEQ 60 ;BR IF YES.
1206 006042 005337 001036 DEC KBPTR ;POINT TO PREVIOUS CHAR.
1207 006046 117716 172764 MOVB #KBPTR,(SP) ;GET PREVIOUS CHAR.
1208 006052 000754 BR 30 ;GO ECHO IT.
1209 006054 005726 60: POPSP
1210 006056 104406 014155 MSGS,ACRLF ;OUTPUT CRLF.
1211 006062 012737 016306 001036 70: MOV #KBUF,KBPTR ;RESET KBUF POINTER.
1212 006070 000751 BR 40
1213
1214 ;SERVICE SPACE.
1215
1216 006072 022737 016306 001036 80: CMP #KRUF,KBPTR ;LEADING SPACE?
1217 006100 001741 BEQ 30 ;BR IF YES (IGNORE IT).
    
```

```

1218 006102 000734          BR      28          ;STORE IT.
1219
1220          ;SERVICE CR AND LF.
1221
1222 006104 112677 172726    98:    MOVB   (SP)+,0KBPTR ;STORE CHAR.
1223 006110 012737 016306    001036  MOV    #KBUF,KBPTR   ;POINT TO START OF KBUF.
1224 006116 104406 014155          MSG#,ACRLF           ;OUTPUT CRLF
1225 006122 104401 007230    000000  QUE#,DECODE,0       ;QUE TO SERVICE COMMAND.
1226
1227          ;SERVICE KBUF OVERFLOW.
1228
1229 006130 104406 014416    128:  MSG#,KBOFLO          ;TYPE KBUF OFLO.
1230 006134 000752          BR      78
    
```

```

1231          ;SERVICE CTRL C, ENDS RUN MODE ALSO.
1232          ;DROPS TO BANKS 0 AND 1 BEFORE RETURNING (IF KT11 IN USE)
1233          ;STACK POINTER SHOULD BE SETUP AHEAD OF CALL TO CTRLX TO PRESERVE
1234          ;NEEDED INFORMATION PRIOR TO REMAPPING TO 0, OR CAUSE REMAPPING TO
1235          ;CURRENT AREA AFTER
1236
1237 006136 012706 016546    CTRLCA: MOV    #SPBOT,SP ;REINITIALIZE STACK POINTER
1238 006142 004737 006230    CTRLCB: JSR    PC,CTRLX  ;CLEAR QUEUES, TYPE 'C
1239 006146 005037 001224          CLR    OFFSTD          ;GET PROGRAM BACK TO BANK 0
1240 006152 004737 014060          JSR    PC,MVCODE
1241 006156 105737 001257          TSTB  RMODE           ;IN RUN MODE?
1242 006162 001002          BNE   18              ;BR IF YES.
1243 006164 000137 007374          JMP    COMCO3         ;BACK TO KYBD ROUTINE.
1244 006170 105037 001257    18:    CLR    RMODE          ;CLEAR RUN MODE INDICATOR
1245 006174 105737 001260          TSTB  CHN             ;IN CHAIN MODE?
1246 006200 100411          BMI   CTRLCD          ;BR IF YES, BYPASS SUMMARY AND RETURN TO
1247          ;CHAIN MONITOR
1248 006202 052777 000100    172736  BIS    #IE,ATKS       ;REFRABLE KEYBOARD INTERRUPTS
1249 006210 104406 014136          MSG#,SUMMARY          ;TYPE RUN END SUMMARY TITLE.
1250 006214 104406 014125    26:    MSG#,CR             ;CARRIAGE RETURN
1251 006220 004737 010274          JSR    PC,DIRA        ;TYPE RUN SUMMARY.
1252 006224 000137 012722    CTRLCD: JMP    CHNOUT    ;EXIT, OR RETURN TO KYBD RTN.
1253
    
```

```

1254 006230 012746 000340          CTRLX: MOV    #PRTY7,-(SP) ;SETUP NEW PSW ON STACK
1255 006234 012746 006242          MOV    #48,-(SP) ;PUT RETURN PC ON STACK
1256 006240 000002                    RTI                    ;LOAD NEW PSW AND CONTINUE
1257 006242 012737 000144 001374 48: MOV    #100,,CSTART ;CLEAR QUEUES AND DELAY TOO.
1258
1259 006250 105737 001257                    TSTB   RMODE ;IF NOT IN RMODE, SHORTEN WAIT
1260 006254 001003                    BNE    1$
1261 006256 012737 000010 001374      MOV    #10,CSTART
1262 006264 004737 012272 18:     JSR   PC,CLRQUS ;CLEAR QUEUES,
1263 006270 005337 001374          DEC   CSTART ;DONE?
1264 006274 001373                    BNE    1$ ;BR IF NOT.
1265 006276 011604                    MOV    (SP),R4 ;SAVE RETURN PC
1266 006300 000005                    RESET
1267 006302 010416                    MOV    R4,(SP) ;RESTORE RETURN PC IN CURRENT BANK
1268 006304 004737 012272          JSR   PC,CLRQUS ;CLEAR QUEUES IN BANK 0
1269 006310 005046 38:     CLR   -(SP) ;RETYPE NEW PSW ON STACK
1270 006312 012746 006320          MOV    #58,-(SP) ;PUT RETURN PC ON STACK
1271 006316 000002                    RTI                    ;LOAD NEW PSW AND CONTINUE
1272 006320 105237 001013 58:     INCH  BRAKE ;SET BRAKE TO PREVENT BACKGROUND MODULE FROM BEING START
1273 006324 013737 001266 001264      MOV    ROTNUM,ROTCNT ;RESET THE ROTATION COUNTER
1274 006332 104406 014153          MSG$,CTRLC ;OUTPUT "C
1275 006336 000207          RTS    PC ;EXIT.
    
```

```

1276 ;SUBROUTINE TO TYPE MULTIPLE ASCII STRINGS, NOTE THAT THE PREVIOUS
1277 ;VALUE OF R5 IS NOT SAVED.
1278
1279 006340 005726          TYPE: TST   (SP)+ ;REMOVE OLD P5 FROM STACK
1280 006342 012546          TYPE: MOV   (R5)+,-(SP) ;GET ASCII ADDRESS.
1281 006344 001403                    BEQ    1$ ;BR IF 0 TERMINATOR.
1282 006346 004737 006356          JSR   PC,TYPE ;TYPE IT.
1283 006352 000773                    BR    TYPE ;GET MORE.
1284 006354 000205 18:     RTS    R5 ;RETURN.
    
```

```

1285 ;TYPE SUBROUTINE.
1286 ;TYPE$ ASCII$ STRING
1287 ;NOTE THAT IF RMODE IS SET BUT MODULES AREN'T YET BEING STARTED, BRAKE
1288 ;MUST BE SET TO PREVENT STARTNG MODULES WHILE TYPING
1289
1290 006356 012737 006426 001410 TYPE: MOV #TTSRV1,TTYLNK
1291 006364 012637 001106 MOV (SP)+,TYPRET ;SAVE RETURN ADDRESS
1292 006370 012637 001104 MOV (SP)+,TYPADR ;SAVE MESSAGE ADDRESS
1293 006374 010137 001124 MOV R1,TYPR1 ;SAVE REGISTERS 1-5
1294 006400 012701 001126 MOV #TYPR2,R1
1295 006404 010221 MOV R2,(R1)+
1296 006406 010321 MOV R3,(R1)+
1297 006410 010421 MOV R4,(R1)+
1298 006412 010521 MOV R5,(R1)+
1299 006414 012777 000100 172530 MOV #100,@TPS ;SET IE
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320 006422 000137 001702 JMP QUETST
1321 006426 117746 172452 TTSRV1: MOV @TYPADR,-(SP) ;GET CHAR.
1322 006432 001015 BNE TYPEB ;BR IF NOT TERMINATOR.
1323 006434 005726 TST (SP)+
1324 006436 012701 001126 MOV #TYPR2,R1 ;RESTORE REGISTERS 1-5
1325 006442 012102 MOV (R1)+,R2
1326 006444 012103 MOV (R1)+,R3
1327 006446 012104 MOV (R1)+,R4
1328 006450 012105 MOV (R1)+,R5
1329 006452 013701 001124 MOV TYPR1,R1
1330 006456 005077 172470 CLR @TPS ;CLEAR IE
1331 006462 000177 172420 JMP @TYPRET
1332
1333
1334 006466 062737 000001 001104 TYPE: ADD #1,TYPADR
1335 006474 122716 000045 CMPB #45,(SP) ;IS IT #?
1336 006500 001037 BNE TYPEL ;NO- BRANCH
1337 006502 112716 000015 TYPEC: MOV #15,(SP) ;OUTPUT CR.
1338 006506 012737 006516 001410 MOV #16,TTYLNK
1339 006514 000431 BR TYPEL
1340 006516 112746 000912 10: MOV #12,-(SP) ;OUTPUT LF.
    
```

```

1341
1342
1343
1344 006522 012737 006532 001410 MOV #26,TTYLNK
1345 006530 000423 BR TYPEL
1346 006532 113737 001254 001020 28: MOVB FILCNT,FILCTR ;GET FILL COUNT.
1347 006540 001002 BNE TYPEE ;BR IF NOT 0.
1348 006542 105237 001020 INCB FILCTR ;OOPS, MAKE IT A 1.
1349 006546 113746 001255 TYPEE: MOV FILLER,-(SP) ;OUTPUT FILLER.
1350 006552 012737 006562 001410 TYPED: MOV #36,TTYLNK
1351 006560 000407 BR TYPEL
1352 006562 105337 001020 38: DECB FILCTR ;DECREMENT FILL COUNTER.
1353 006566 001367 BNE TYPEE ;BR IF NOT 0.
1354 006570 012737 006426 001410 MOV #TTSRV1,TTYLNK
1355 006576 000713 BR TTSRV1
1356 006600 012677 172350 TYPEL: MOV (SP)+,@TPB
1357 006604 012737 010000 001110 MOV #10000,TYPCTR
1358 006612 000137 001702 JMP QUETST
    
```

```

;ROUTINE TO SERVICE CKDATA CALL (CDATA6)
1359
1360
1361 006616 112737 177777 001315 DATCK,i MOVB #=1,DATFLG ;SET FLAG FOR DATACK CALL
1362 006624 000402 BR CKCONT ;CONTINUE
1363 006626 105037 001315 CDATA,i CLR B DATFLG ;SET FLAG FOR CKDATA CALL
1364
1365 006632 005037 001142 CKCONT,i CLR CDPLG
1366 006636 005037 001114 CLR DERRCT ;INITIALIZE DATA ERROR COUNT
1367 006642 004437 012244 JSR R4,SAV04 ;TEMPORARILY SAVE REGS
1368 006646 016604 000012 MOV 12(SP),R4 ;GET PC+2 OF CALL
1369 006652 010437 001400 CDATA1,i MOV R4,CDERR1 ;SETUP PC OF CALL
1370 006656 162737 000002 001400 SUB #2,CDERR1
1371 006664 012401 MOV (R4)+,R1 ;GET BEGIN ADDRESS OF MODULE
1372 006666 012403 MOV (R4)+,R3 ;GET TABLE ADDRESS OF RBUFFA FROM MODULE
1373 006670 010361 000052 MOV R3,SBADR(R1) ;SAVE IT
1374 006674 062701 000032 ADD #SVR0,R1 ;SAVE MODULE'S REGISTERS
1375 006700 012621 MOV (SP)+,(R1)+
1376 006702 012621 MOV (SP)+,(R1)+
1377 006704 012621 MOV (SP)+,(R1)+
1378 006706 012621 MOV (SP)+,(R1)+
1379 006710 012621 MOV (SP)+,(R1)+
1380 006712 010511 MOV R5,(R1)
1381 006714 022626 CMP (SP)+,(SP)+
1382 006716 016401 177774 MOV -4(R4),R1 ;RESTORE MODULE ADDR
1383 006722 012300 MOV (3)+,R0 ;GET READ BUFFER PA
1384 006724 005723 TST (3)+ ;SKIP EA BITS
1385 006726 012305 MOV (3)+,R5 ;BUFFER WORD COUNT
1386 006730 016102 000074 MOV WBUFFA(R1),R2 ;GET WRITE BUFFER PA
1387 006734 005003 CLR R3 ;CLEAR OUT REG. 3
1388 006736 005737 001142 TST CDPLG ;ENTERED FROM DATA ERROR RETURN?
1389 006742 001413 BEQ 308 ;NO- BRANCH
1390 006744 012603 MOV (SP)+,R3 ;YES- GET COUNT
1391 006746 005203 INC R3
1392 006750 006303 ASL R3 ;CREATE OFFSET
1393 006752 060300 ADD R3,R0 ;GET ADDRESS OF CURRENT READ BUFFER WORD
1394 006754 103003 BCC ,+10
1395 006756 060302 ADD R3,R2 ;GET ADDRESS OF CURRENT WRITE BUFFER WORD
1396 006760 103003 BCC ,+10
1397 006762 006203 AGR R3 ;RESTORE COUNT
1398 006764 160305 SUB R3,R5 ;CALCULATE WORDS LEFT
1399 006766 012637 001114 MOV (SP)+,DERRCT ;RESTORE DATA ERROR COUNT
1400 006772
1401 006772 021012 308i CMP #R0,#R2 ;DATA OK?
1402 006774 001427 18i BEQ 38 ;YES, BRANCH
1403 006776 005237 001114 INC DERRCT ;NO, COUNT DATA ERROR
1404 007002 023727 001114 000003 CMP DERRCT,#3 ;MORE THAN 3 ERRORS?
1405 007010 003404 BLE 28 ;NO- REPORT ERROR
1406 007012 032777 002000 172136 BIT #BIT10,#SR ;YES- REPORT ALL?
1407 007020 001415 REQ 38 ;NO- BRANCH
1408 007022 011061 000060 28i MOV #R0,#WAS(R1) ;SAVE ACTUAL DATA
1409 007026 011261 000056 MOV #R2,#ASB(R1) ;SAVE EXPECTED DATA
1410 007032 010361 000054 MOV R3,#WASADR(R1) ;SAVE COUNT
1411 007036 013746 001114 MOV DERRCT,-(SP) ;SAVE DATA ERROR COUNT
1412 007042 010346 MOV R3,-(SP) ;SAVE WORD COUNT ON STACK
1413 007044 013746 001400 MOV CDERR1,-(SP) ;SAVE VA OF CKDATA CALL
1414 007050 000137 001376 JMP CDERR ;GO TO ERROR CALL
    
```

```

1415 007054 005203 38i INC R3 ;COUNT WORDS CHECKED
1416 007056 005720 TST (R0)+
1417 007060 005722 TST (R2)+
1418 007062 005305 58i DEC R5
1419 007064 001342 BNE 18
1420 007066 005737 001114 TST DERRCT ;ANY DATA ERRORS?
1421 007072 001003 BNE 68 ;YES, BRANCH
1422 007074 012437 001054 MOV (R4)+,DSTADR ;GET THE RETURN ADDRESS
1423 007100 000423 BR 78 ;GO TO COMMON RETURN CODE
1424 007102 005261 000030 68i INC ERCNT(R1) ;DATA ERRORS OCCURRED- UP MODULE ERROR
1425 ;COUNT BY 1
1426 007106 001002 BNE ,+6
1427 007110 005161 000030 000060 COM ERCNT(R1) ;SET TO -1 IF 0
1428 007114 013761 001114 MOV DERRCT,#WAS(R1) ;STORE INFORMATION IN MODULE
1429 007122 010361 000056 MOV R3,#ASB(R1) ;STORE SIZE IN ASB
1430 007126 010446 MOV R4,-(SP) ;SAVE POINTER TO ERROR RETURN ADDRESS
1431 007130 010661 000046 MOV SP,#SVR6(R1) ;SAVE STACK POINTER
1432 007134 104422 MSGD# ;REPORT TOTALS, HALT MODULE IF NEEDED
1433 007136 012604 MOV (SP)+,R4 ;GET ERROR RETURN POINTER
1434 007140 013701 001122 MOV MODADR,R1 ;GET MODULE START ADDRESS
1435 007144 012437 001054 MOV (R4)+,DSTADR ;GET THE RETURN ADDRESS
1436 007150 105737 001315 78i TST BPL DATFLG ;IS THIS A DATACK CALL ?
1437 007154 100002 BPL 88 ;NO, CONTINUE
1438 007156 013734 001114 MOV DERRCT,#(R4)+ ;YES, LOAD THE NUMBER OF DATA ERRORS
1439 007162 116137 000020 001066 88i STAT(R1),RSTAT ;SET UP FOR MODULE RETURN
1440 007170 010100 MOV R1,R0
1441 007172 062700 000046 ADD #SVR6,R0
1442 007176 000137 002570 JMP IOGSVD
1443
1444 007202 005746 CDRET: TST -(SP) ;PUSH STACK
1445 007204 004437 012244 JSR R4,SAV04
1446 007210 016604 000014 MOV 14(SP),R4 ;GET VA OF CKDATA CALL
1447 007214 062704 000002 ADD #2,R4
1448 007220 005237 001142 INC CDPLG
1449 007224 000137 006652 JMP CDATA1
    
```

```

1450 ;COMMAND DECODER. CHECKS SYNTAX AGAINST COMMAND TABLE.
1451 ;ALL COMMANDS ARE CHECKED AGAINST THE QUOTES. NO ABBREVIATIONS ALLOWED
1452
1453 007230 105737 001010 DECODE: TSTB SPCFLG ;SPECIAL FLAG SET?
1454 007234 001404 BEQ 18 ;BR IF NOT.
1455 007236 005037 001010 CLR SPCFLG ;YES, CLEAR IT.
1456 007242 000177 171610 JMP #SPETRN ;GO VIA SRETRN.
1457 007246 105737 001257 16: TSTB RMODE ;IN RUN MODE?
1458 007252 001403 BEQ 28 ;NO, CONTINUE
1459 007254 104406 014304 MSG6,INVCD1 ;'INVALID CMD. ONLY CNTL C IS OK'
1460 007260 000441 BR COMCON ;CONTINUE
1461 007262 004437 012244 26: JSR R4,SAV04 ;SAVE REGS 0-4.
1462 007266 013700 001036 MOV KBPTR,R0 ;POINTER TO R0.
1463 007272 012701 015232 GTOK: MOV #COMTAB,R1 ;DEVICE DECODING COMES HERE
1464 007276 010046 GTOKX: MOV R0,=(SP) ;SAVE STRING POINTER
1465 007300 122021 GTOK1: CMPB (R0)+,(R1)+ ;MATCH THE CHARACTER?
1466 007302 001407 BEQ GTOK2 ;YEP, SEE IF THE REST MATCH
1467 007304 105721 TSTB (R1)+ ;NO, GO CHECK AGAINST NEXT
1468 007306 001376 BNE =-2 ;BUT FIND WHEPE NEXT ENTRY IS
1469 007310 111102 MOVB (R1),R2 ;DISPLACEMENT TO NEXT ENTRY
1470 007312 001422 BEQ COMC01 ;NO SUCH THING.
1471 007314 060201 MORE: ADD R2,R1 ;POINT TO NEXT ENTRY
1472 007316 011600 MOV (SP),R0 ;POINT TO BEGINNING OF INPUT
1473 007320 000767 BP GTOK1 ;KEEP LOOKING
1474 007322 105711 GTOK2: TSTB (R1) ;LAST CHR IN ENTRY?
1475 007324 001365 BNE GTOK1 ;NO,CHECK THE OTHER CHR'S
1476 007326 122126 CMPB (R1)+,(SP)+ ;RESET SP AND R1
1477 007330 111102 MOVB (R1),R2 ;GET DISPATCH ADDRESS
1478 007332 060201 ADD R2,R1 ;R1 POINTS TO NEXT ENTRY
1479 007334 014137 001372 MOV =(R1),CADDR
1480 007340 005037 001374 CLR CSTART ;SET MODULE ADDR IN CSTART TO 0.
1481 007344 010037 001036 MOV R0,KBPTR ;SAVE CURRENT BUFFER POINTER.
1482 007350 004737 012256 JSR PC,RST04 ;RESTORE REGS 0-4.
1483 007354 000137 001362 JMP COMQUE ;GO TO COMMON QUE CALL.

```

```

1484 ;COMMON RETURN FROM COMMAND EXECUTION AND RUN ROUTINE (INCLUDING
1485 ;RELOCATION AND INTERNAL RESTART)
1486
1487 007360 104406 014272 COMC01: MSG6,INVCD ;TYPE INVALID COMMAND.
1488 007364 COMCON:
1489 007364 005037 001010 COMC02: CLR SPCFLG ;CLEAR SPECIAL FLAG AND
1490 CLR DIR ;DIR COMMAND INDICATOR.
1491 007370 105037 001272 CLR FB ;CLEAR FILL COMMAND INDICATOR.
1492 007374 104406 014122 COMC03: MSG6,DOT ;TYPE DOT.
1493 007400 000137 001650 COMC04: JMP INPUT ;GO GET MORE INPUT.

```

```

1494          ;SPECIAL INPUT ROUTINE.
1495          ;CALLED BY MODIFY ROUTINE, WHEN WAITING FOR NEW VALUE TO BE INPUT.
1496
1497 007404 012637 001056 SINPUT: MOV    (SP)+,SRETRN  ;SAVE RETURN ADDR.
1498 007410 105237 001010        INCB   SPCFLG    ;SET SPECIAL FLAG.
1499 007414 000771          BR     COMCO4

```

```

1500          ;RUN ROUTINE. STARTS EXERCISER EXECUTION.
1501          ;NOTE THAT "RUN" IS ENTERED ONLY FROM BANK 0, BUT "RUNRES" IS USED WHEN
1502          ;THE PROGRAM IS NOT IN BANK 0.
1503
1504 007416 000402          BR     ;+6
1505 007420 105037 001275 RUN:  CLRR   LOCK          ;CLEAR LOCK TO ALLOW CODE RELOCATION
1506 007424 005037 001224        CLR    OFFSTD        ;
1507 007430 005037 001244        CLR    SYSCNT        ;ZERO SYSTEM ERROR COUNTER
1508 007434 005037 001246        CLR    PWRCNT        ;ZERO POWER FAIL COUNTER
1509 007440 012737 042761 010470 28:  MOV    #42761,SLDSR  ;RESTORE THESE INSTRUCTIONS
1510 007446 012737 042761 010512        MOV    #42761,SLDSE  ;FOR A CHECK SUM MODULE
1511 007454 105037 001274        CLRB   RUNPFL        ;
1512 007460 105037 001257        CLRR   RMODE        ;
1513 007464 012737 016234 001250        MOV    #MODC-2,BKPTR  ;
1514 007472 004737 012354        JSR    PC,CLRVEC    ;INITIALIZE TRAPCATCHER IN VECTOR AREA
1515 007476 004737 012272        JSR    PC,CLRQUS    ;INCLUDES CLEARING RPAR,MODCNT,IOBKID
1516 007502 112737 177777 001023 118:  MOVB   #=1,#BFLG    ;INITIALIZE BUFFER ROTATION FLAG
1517 007510 012702 016236        MOVB   #MODC,R2     ;SETUP TO COUNT SELECTED MODULES, AND
1518          ;IF NOT SYSERR OR POWER FAIL RESTART
1519          ;CLEAR MODULES' PASCNT AND ERRCNT.
1520 007514 012201          16:  MOV    (2)+,R1      ;MODULE ADDR TO R1.
1521 007516 001462          BEQ    RUNC        ;RR IF NO MORE.
1522 007520 105737 001257        TSTB   PMODE        ;INTERNAL RESTART?
1523 007524 001012          BNE    28          ;RR IF YES.
1524 007526 105737 001260        TSTB   CHW          ;IN CHAIN MODE?
1525 007532 100407          BMI    28          ;BR IF YES.
1526 007534 005061 000026        CLR    PSCNT(R1)   ;CLEAR MOD'S PASCNT.
1527 007540 005061 000030        CLR    ERCNT(R1)   ;CLEAR MOD'S ERROR COUNT.
1528 007544 042761 020000 000020        RIC   #BIT13,STAT(R1) ;CLEAR STOPPED BIT.
1529 007552 032761 040000 000020 28:  BIT   #BIT14,STAT(R1) ;MODULE SELECTED?
1530 007560 001755          BEQ    18          ;BR IF NOT.
1531 007562 032761 020000 000020        BIT   #BIT13,STAT(R1) ;MODULE STOPPED?
1532 007570 001351          BNE    18          ;BR IF YES.
1533 007572 005737 001222        TST   OFFSET        ;IF OFFSET 0, IGNORE BITS 10+11
1534 007576 001414          BEQ    38          ;
1535 007600 032761 004000 000020        BIT   #BIT11,STAT(R1) ;NOT 0- IF BIT11 IS SET, DON'T COUNT IT
1536 007606 001342          BNE    18          ;
1537 007610 032737 001777 001222        RUNC   #1777,OFFSET ;32K BOUNDARY?
1538 007616 001404          BEQ    38          ;YES- IGNORE BIT 10
1539 007620 032761 020000 000020        BIT   #BIT10,STAT(R1) ;NO- BIT 10 SET?
1540 007626 001332          BNE    18          ;YES- DON'T RUN THIS MODULE
1541 007630 105237 001015 38:  INCB   MODCNT        ;NO, UP COUNT OF RUNABLE MODULES.
1542 007634 005761 000020        TST   STAT(R1)     ;RKMDD?
1543 007640 100402          BMI    ;+6        ;NO- BRANCH
1544 007642 105237 001025        INCB   BKCNT        ;YES- UP COUNT
1545 007646 032761 010000 000020        BIT   #BIT12,STAT(R1) ;EXTENDED MODULE?
1546 007654 001402          BEQ    ;+6        ;NO- BRANCH
1547 007656 105237 001026        INCB   XCNT        ;YES- COUNT IT
1548 007662 000714          BR     18          ;GO CHECK NEXT MODULE.
1549 007664 105737 001015 RUNC:  TSTB   MODCNT        ;ANY RUNABLE MODULES?
1550 007670 001005          BNE    58          ;YES, BRANCH
1551 007672 005737 001222        TST   OFFSET        ;NO, OFFSET 0?
1552 007676 001501          BEQ    RUND        ;YES- COMMAND INVALID
1553 007700 000137 004604        JMP    ENDSVB      ;NO- THE ONLY MODS RUNNABLE MUST BE THOSE
1554          ;WITH BITS 10 OR 11 SET- RELOCATE
1555 007704 113737 001015 001016 58:  MOVB   MODCNT,MODCTR ;LOAD COUNT OF RUNNABLE MODULES

```

```

1556 ;INTO STARTUP COUNTER
1557 007712 012737 016234 001040 MOV #MODQ-2,MODPTR ;MODULE TABLE ADDR TO MODPTR,
1558 007720 012737 040000 001060 MOV #40000,IOBKID ;START WITH SPECIAL BACKGROUND MODULES,
1559 007726 105237 001320 INCB NSTOP ;SET THE NON-STOP FLAG
1560 007732 105737 001261 TSTB RDTI
1561 007736 001447 BEQ 2$
1562 007740 113737 001026 001031 MOV# XCNT,ROTCT ;SETUP ROTATION TIMEOUT COUNTER- 16 TIMES NUMBER
1563 ;OF EXTENDED MODULES
1564 007746 106337 001031 ASLB ROTCT
1565 007752 106337 001031 ASLB ROTCT
1566 007756 106337 001031 ASLB ROTCT
1567 007762 106337 001031 ASLB ROTCT
1568 007766 123727 001026 000020 CMPB XCNT,#20 ;PREVENT LOSS OF COUNT IF OVERFLOW
1569 007774 103403 BLO 1$
1570 007776 112737 000377 001031 MOV# #377,ROTCT
1571 010004 113737 001031 001302 1$: MOV# ROTCT,#GWBFC ;INITIALIZE WRITE BUFFER ROTATION COUNTER
1572 010012 113737 001015 001033 MOV# MODCNT,PASREL ;SET ALTERNATE RELOCATION COUNT TO 8
1573 010020 106337 001033 ASLA PASREL ;TIMES MODCNT
1574 010024 106337 001033 ASLB PASREL
1575 010030 106337 001033 ASLB PASREL
1576 010034 123727 001015 000040 CMPB MODCNT,#40
1577 010042 103403 BLO 3$
1578 010044 112737 000377 001033 MOV# #377,PASREL
1579 010052 004737 010126 3$: JSR PC,SETWBF
1580 010056 105737 001257 2$: TSTB RMODE
1581 010062 001005 BNE 4$
1582 010064 104406 014122 MSGS,DOT
1583 010070 112737 000001 001257 MOV# #1,RMODE ;SET RMODE IN CURRENT BANK
1584 010076 000137 001650 4$: JMP INPUT
1585 010102 005037 001224 RUND: CLR OFFSTD
1586 010106 004737 014060 JSR PC,MVCODE
1587 010112 105037 001257 CLR# RMODE
1588 010116 104406 014346 MSGS,INVCD4 ;'NO MODULES SELECTED'
1589 010122 000137 007364 JMP COMCON
    
```

```

1590 ;SETUP WRITE BUFFER INFORMATION
1591
1592 010126 005037 000076 SETWBF: CLR WBUFEA ;ZERO EA BITS
1593 010132 013737 001200 001212 MOV LCORE,WBUF ;LOAD STARTING ADDR OF WRITE BUFFER
1594 010140 013737 001204 001216 MOV LCOPI2,WBUF12
1595 010146 063737 001222 001216 ADD OFFSET,WBUF12
1596 010154 023737 001216 001206 CMP WBUF12,HCOP12 ;IS WRITE BUFFER AT THE TOP ?
1597 010162 001001 BNE SETWB1 ;NO, IT'S OK, CONTINUE
1598 010164 000207 RTS ;RETURN
1599
1600 010166 013737 001206 001236 SETWB1: MOV HCOPI2,MXWBF ;CALCULATE MAX SIZE OF WRITE BUFFER
1601 010174 105737 001260 TSTB CHN ;IN CHAIN MODE ?
1602 010200 100003 BPL 1$ ;NO, DON'T ADD ANY MORE WRITE BUFFER
1603 010202 062737 000040 001236 ADD #40,MXWBF ;YES, ADD 1K TO MAX. WRITE BUFFER SIZE
1604 010210 163737 001216 1$: SUB WBUF12,MXWBF
1605 010216 006337 001236 ASL MXWBF
1606 010222 006337 001236 ASL MXWBF
1607 010226 006337 001236 ASL MXWBF
1608 010232 006337 001236 ASL MXWBF
1609 010236 006337 001236 ASL MXWBF
1610 010242 103003 BCC .+10
1611 010244 012737 177777 001236 MOV #177777,MXWBF
1612 010252 000207 RTS PC
    
```

```

1613                                     ;MAP ROUTINE. TYPES RESIDENT MODULES AND THEIR START ADDRESS.
1614
1615 010254 105237 001011      MAP:   INCB   DIRIND   ;SET DIR INDICATOR.
1616 010260 105037 015450      CLRB   MDIRE    ;TERMINATE ASCII STRING EARLY.
1617 010264 004737 010274      JSR    PC,DIRA ;TYPE MAP.
1618 010270 000137 007364      JMP    CONCON
1619
1620 010274 012702 016236      DIRA:  MOV    #MODQ,R2 ;GET MODULE TABLE ADDR.
1621 010300 012201 016236      18:   MOV    (2)+,R1 ;GET MODULE ADDR.
1622 010302 001446 016236      BEQ    58 ;BR IF 0. ALL DONE.
1623 010304 032761 040000 000020  BIT    #BIT14,STAT(R1) ;MODULE SELECTED?
1624 010312 001003 016236      BNE    24 ;BR IF YES.
1625 010314 105737 001011      TSTB  DIRIND   ;TYPING DIRECTORY?
1626 010320 001767 016236      BEQ    18 ;BR IF NOT. DONT TYPE UNSELECTED MODS.
1627 010322 004537 011206      28:   JSR    R5,FILLNM ;FILL MOD NAME IN ASCII STRING.
1628 010326 015415 016236      AMODNM+1 ;ADDR TO STUFF NAME IN.
1629 010330 010146 016236      MOV    R1,-(SP) ;MODULE ADDR TO STACK.
1630 010332 004537 011512      JSR    R5,OACNV ;CONVERT MOD ADDR TO ASCII.
1631 010336 015426 016236      APC
1632 010340 016146 000020      MOV    STAT(R1),-(SP) ;CONVERT MODULE STATUS.
1633 010344 004537 011512      JSR    R5,OACNV
1634 010350 015442 016236      AMDSTA
1635 010352 016146 000026      MOV    PSCNT(R1),-(SP) ;MOD'S PASS COUNT TO STACK.
1636 010356 004537 011744      JSR    R5,BDCNV ;CONVERT IT TO DECIMAL ASCII.
1637 010362 015461 016236      APSCNT
1638 010364 016146 000030      MOV    ERCNT(R1),-(SP) ;MOD'S EPROR COUNT TO STACK.
1639 010370 004537 011744      JSR    R5,BDCNV ;CONVERT IT TO DECIMAL ASCII.
1640 010374 015500 016236      AERRS
1641 010376 105737 001011      TSTB  DIRIND   ;TYPING DIRECTORY?
1642 010402 001003 016236      BNE    38 ;BR IF YES.
1643 010404 112737 000040 015450  MOVB  #40,MDIRE ;NO. ALLOW TYPING FULL STRING.
1644 010412 104406 015414      38:   MSGS,AMODNM ;TYPE ASCII LINE.
1645 010416 000730 016236      BR     18 ;DO IT AGAIN.
1646 010420 000207 016236      58:   RTS    PC ;EXIT.

```

```

1647                                     ;SELECT MODULE(S) ROUTINE.
1648
1649 010422 012737 052761 010470  SEL:   MOV    #52761,SLDSB ;MODIFY COMMON TO SELECT MODULE(S).
1650 010430 012737 052761 010512  MOV    #52761,SLDSE ;
1651 010436 000406 016236      BR     SLDS ;GO TO COMMON.
1652
1653                                     ;DESELECT MODULE(S) ROUTINE.
1654
1655 010440 012737 042761 010470  DES:   MOV    #42761,SLDSB ;MODIFY COMMON TO DSELECT MODULE(S).
1656 010446 012737 042761 010512  MOV    #42761,SLDSE ;
1657
1658                                     ;SELECT/DESELECT COMMON.
1659
1660 010454      SLDS:
1661 010454 004737 011234      SLDSA: JSR    PC,GETNAM ;CHECK NAME. GET MODULE ADDR.
1662 010460 000410 016236      BR     SLDSC ;NO NAME. SELECT/DESELECT ALL.
1663 010462 000405 016236      BR     SLDSL ;INVALID OR NEX NAME.
1664 010464 013701 001040      MOV    WDPTR,R1 ;MODULE ADDR TO R1.
1665 010470 042761 040000 000020  SLDSB: BIC    #BIT14,STAT(R1) ;SELECT/DESELECT MODULE.
1666 010476 000137 007364      SLDSL: JMP    COMCON ;DONE.
1667 010502 012702 016236      SLDSC: MOV    #MODQ,R2 ;MODULE TABLE ADDR TO R2.
1668 010506 012201 016236      SLDSL: MOV    (2)+,R1 ;GET MODULE ADDR.
1669 010510 001772 016236      BEQ    SLDSL ;BR IF 0. END OF TABLE. DONE.
1670 010512 042761 040000 000020  SLDSE: BIC    #BIT14,STAT(R1) ;SELECT/DESELECT MODULE.
1671 010520 000772 016236      BR     SLDSL ;DO IT AGAIN.
1672                                     ;LOCATIONS SLDSB AND SLDSE ARE PURE WHILE IN RUN MODE.

```

```

1673 010522          PON:
1674 010522          POFF:
1675 010522          KTON:
1676 010522 000137 007374 KTOFF: JMP      COMC03      ;IGNORE THE COMMAND
1677
1678                  ;THIS ROUTINE ENABLES WRITE BUFFER ROTATION
1679
1680 010526 105037 001262 ROTON: CLRB   ROTL      ;CLEAR ROTATION LOCK FLAG
1681 010532 004737 013262      JSR    PC,ROTCHK    ;GO DO "ENABLED" MESSAGE
1682 010536 000137 007374      JMP    COMC03      ;CONTINUE
1683
1684
1685                  ;THIS ROUTINE DISABLES WRITE BUFFER ROTATION
1686
1687 010542 112737 000377 001262 ROTOFF: MOVB  #377,ROTL ;SET THE ROTATION LOCK FLAG
1688 010550 013737 001266 001264      MOV   ROTNUM,ROTCNT ;LOAD THE ROTATION COUNTER
1689 010556 104406 014700      MSG#,ROTDIS        ;"MBUF ROTATION DISABLED"
1690 010562 000137 007374      JMP    COMC03      ;CONTINUE
  
```

```

1691                  ;MODIFY ROUTINE.
1692                  ;GETS ADDRESS AND PRINTS ADDR CONTENTS.
1693                  ;USER CAN INPUT NEW NUMBER OR CR.
1694                  ;CR MEANS HE JUST WANTS TO EXAMINE.
1695
1696 010566 112737 177777 001272 FILL: MOVB  #=-1,FILLID ;SET FILL INDICATOR.
1697 010574          MOD:
1698 010574 004737 011234 58:   JSR    PC,GETNAM    ;GET NAME AND MODULE ADDR.
1699 010600 000401          BR     48             ;NO NAME.
1700 010602 000440          BR     38             ;INVALID OR NEX NAME.
1701 010604 004737 011022 46:   JSR    PC,GETADR    ;GET ADDRESS.
1702 010610 000430          BR     68             ;INVALID ADDR OR NO ADDR.
1703 010612 013704 001042      MOV   ADDRLO,R4      ;MOVE ADDR TO R4.
1704 010616 063704 001040      ADD  MODPTR,R4      ;ADD MODULE ADDR TO ADDR.
1705 010622 010446          MOV   R4,-(SP)      ;TYPE ADDR.
1706 010624 004737 011000      JSR   PC,Itoa       ;
1707 010630 011446          MOV   (R4),-(SP)    ;GET CONTENTS OF ADDR.
1708 010632 004737 011000      JSR   PC,Itoa       ;PRINT IT OUT.
1709 010636 004737 007404      JSR   PC,SINPUT     ;GET NEW CONTENTS.
1710 010642 004737 011070      JSR   PC,GETNUM     ;CONVERT INPUT TO BINARY.
1711 010646 000416          BR     38             ;INVALID DATA (NOT A NUMBER).
1712 010650 005737 001062      TST  YES            ;NUMBER?
1713 010654 001401          BEQ  28             ;BR IF NOT.
1714 010656 010114          MOV  R1,(R4)        ;STORE NEW VALUE.
1715 010660 122702 000012 26:   CMPB #12,R2         ;LINE FEED?
1716 010664 001007          BNE  36             ;NO, DONE.
1717 010666 005724          TST  (R4)+          ;POINT TO NEXT LOCATION.
1718 010670 000754          BR   16             ;REPEAT FOR NEXT LOC.
1719 010672 012704 001254 68:   MOV  #FILCNT,R4     ;GET FILCNT ADDR.
1720 010676 105737 001272      TSTB FILLID        ;IN FILL MODE?
1721 010702 001352          BNE  78             ;BR IF YES.
1722 010704 000137 007364 38:   JMP  COMCON        ;DONE.
  
```



```

1748 ;THIS ROUTINE GETS THE ADDRESS ASSOCIATED WITH THE MOD COMMAND
1749
1750 011022 004737 011070 GETADR: JSR PC,GETNUM ;GET NUMBER
1751 011026 000414 BR 18 ;INVALID DATA RETURN.
1752 011030 005737 001062 TST YES ;NUMBER?
1753 011034 001411 BEQ 18 ;BR IF NOT.
1754 011036 006201 ASR R1 ;GOOD ADDRESS?
1755 011040 103410 BCS ADRERR ;NOGOOD, IT'S ODD
1756 011042 060101 ADD R1,R1 ;RESTORE THE ADR
1757 011044 010137 001042 MOV R1,ADDRLO ;STORE AT ADDRLO
1758 011050 010337 001044 MOV R3,ADDRHI ;AND ANY HIGH ORDER BITS IN ADDRHI
1759 011054 062716 000002 ADD #2,(SP) ;SET UP GOOD EXIT.
1760 011060 000207 18: RTS PC ;DONE
1761
1762 011062 104406 014364 ADRERR: MSG$,INVADR ;TYPE INVALID ADDR OR DATA.
1763 011066 000207 RTS PC ;EXIT.
    
```

```

1764 ;GET NUMBER FROM KEYBOARD BUFFER, AND RETURN THE LOW ORDER BITS
1765 ;IN R1 AND THE HIGH ORDER BITS IN R3.
1766 ;ROUTINE ADDS 2 TO THE RETURN PC IF SUCCESSFUL.
1767
1768 011070 013700 001036 GETNUM: MOV KBPTR,R0 ;GET BUFFER POINTER.
1769 011074 005001 38: CLR R1 ;DATA
1770 011076 005003 CLR R3
1771 011100 005037 001062 CLR YES ;FLAG
1772 011104 112002 18: MOVB (R0)+,R2 ;GET A BYTE
1773 011106 120227 000040 CMPR R2,#40 ;SPACE?
1774 011112 001770 BEQ 36 ;YES,IGNORE IT
1775 011114 122702 000015 CMPR #15,R2 ;CR?
1776 011120 001425 BEQ 28 ;YES,RETURN
1777 011122 120227 000012 CMPB R2,#12 ;LF?
1778 011126 001422 BEQ 28 ;DONE
1779 011130 120227 000060 CMPR R2,#0 ;LOW LIMIT
1780 011134 002752 BLT ADRERR ;TOO LOW, REPORT THE ERROR
1781 011136 120227 000067 CMPR R2,#7 ;HIGH LIMIT
1782 011142 003347 BGT ADRERR ;TOO HIGH, REPORT THE ERROR
1783 011144 006301 ASL R1 ;SHIFT OLD STUFF
1784 011146 006103 ROL R3
1785 011150 006301 ASL R1 ;3 TIMES LEFT
1786 011152 006103 ROL R3
1787 011154 006301 ASL R1 ;I.E. MULT. BY OCTAL 10
1788 011156 006103 ROL R3
1789 011160 060201 ADD R2,R1 ;ADD NEW TO OLD
1790 011162 162701 000060 SUB #0,R1 ;BUT GET RID OF ASCII STUFF
1791 011166 005237 001062 TNC YES ;SET FLAG
1792 011172 000744 BR 18 ;MORE, MORE
1793 011174 010037 001036 28: MOV R0,KBPTR ;SAVE CURRENT BUF POINTER.
1794 011200 062716 000002 ADD #2,(SP) ;SET UP OK EXIT.
1795 011204 000207 RTS PC ;EXIT.
    
```

1796  
1797  
1798 011206 010346  
1799 011210 012503  
1800 011212 012704 000005  
1801 011216 112123  
1802 011220 005304  
1803 011222 001375  
1804 011224 162701 000005  
1805 011230 012603  
1806 011232 000205

THIS ROUTINE PUTS THE MODULE NAME INTO AN ASCII STRING FOR OUTPUT  
FILLNM: MOV R3,=(SP) ;SAVE R3.  
MOV (R5)+,R3 ;STORE ADDR TO R3.  
MOV #5,R4 ;WILL DO 5 TIMES.  
181: MOVB (R1)+,(R3)+ ;MOVE CHAR.  
DEC R4 ;DONE?  
BNE 18 ;BR IF NOT.  
SUB #5,R1 ;RESTORE R1.  
MOV (SP)+,R3 ;RESTORE R3.  
RTS R5 ;EXIT.

1807  
1808  
1809 011234 013700 001036  
1810 011240 005037 001062  
1811 011244 005037 001040  
1812 011250 012702 000005  
1813 011254 012701 015415  
1814 011260 112721 000040  
1815 011264 005302  
1816 011268 001374  
1817 011270 012701 015415  
1818 011274 111002  
1819 011276 122702 000015  
1820 011302 001442  
1821 011304 122702 000012  
1822 011310 001437  
1823 011312 122702 000040  
1824 011316 001430  
1825 011320 120227 000101  
1826 011324 002403  
1827 011326 120227 000132  
1828 011332 003411  
1829 011334 120227 000060  
1830 011340 002427  
1831 011342 120227 000071  
1832 011346 003024  
1833 011350 005737 001062  
1834 011354 001426  
1835 011356 023727 001062 000005 48:  
1836 011364 002015  
1837 011366 110221  
1838 011370 005237 001062  
1839 011374 005200  
1840 011376 000736  
1841 011400 005737 001062  
1842  
1843  
1844  
1845 011404 001773  
1846 011406 000416  
1847  
1848 011410 005737 001062  
1849 011414 001013  
1850 011416 000405  
1851 011420 104406 014403  
1852 011424 062716 000002  
1853 011430 000207  
1854  
1855  
1856 011432 005037 001062  
1857 011436 010037 001036  
1858 011442 000207  
1859 011444 012702 016236  
1860 011450 012203  
1861 011452 001762  
1862 011454 012701 015415

THIS ROUTINE GETS THE MODULE NAME ASSOCIATED WITH THE MOD AND SEL/DES COMMANDS.  
GETNAM: MOV KBPTR,R0 ;GET BUFFER POINTER.  
CLR YES ;CLEAR SUCCESS INDICATOR.  
CLR MODPTR ;IF 0, ABS ADDR.  
MOV #5,R2 ;WILL DO 5 TIMES.  
MOV #AMODNM+1,R1 ;WILL STORE KYRD NAME IN TMPNAM.  
181: MOVB #40,(R1)+ ;PUT CLEAR IT TO SPACES FIRST.  
DEC R2 ;DONE?  
BNE 18 ;BR IF NOT.  
MOV #AMODNM+1,R1 ;RELOAD TEMP ADDR.  
28: MOVR (R0),R2 ;GET CHAR FROM KBUF AREA.  
CMPB #15,R2 ;CR?  
BEQ 68 ;BR IF YES.  
CMPB #12,R2 ;LF? TREAT AS CR.  
BEQ 68 ;BR IF YES.  
CMPB #40,R2 ;SPACE? DISREGARD LEADING ONES.  
BEQ 58 ;BR IF YES.  
CMPB R2,#'A ;A OR HIGHER?  
BLT 38 ;BR IF LESS, (COULD BE A NUMBER).  
CMPB R2,#'Z ;Z OR LESS?  
BLE 48 ;BR IF Z OR LESS.  
38: CMPB R2,#'0 ;0 OR HIGHER?  
BLT 78 ;BR IF LESS, YOU LOOSE!  
CMPB R2,#'9 ;9 OR LESS?  
RCT 78 ;BR IF GREATER, YOU LOOSE!  
TST YES ;ANY PREVIOUS CHARS?  
BEQ 88 ;BR IF NOT, NOT LEADING NUMERICS ALLOWED!  
48: CNF YES,#5 ;6TH CHARACTER?  
BGE 78 ;IF YES, BRANCH  
MOVB R2,(R1)+ ;STORE CHAR.  
INC YES  
418: INC R0 ;POINT TO NEXT CHAR.  
BR 28 ;DO IT AGAIN!  
58: TST YES ;ANY CHARS?  
BEQ 418 ;BR IF NOT.  
BR 98  
68: TST YES ;ANY CHARS?  
BNE 98 ;BR IF YES.  
BR 88 ;NO NAME.  
78: MSG8,INVNAM ;TYPE INVALID OR NEX (NON-EXISTENT) MESSAGE.  
ADD #2,(SP) ;SET UP INVALID NAME EXIT.  
RTS PC ;EXIT.  
88: CLR YES ;  
MOV R0,KBPTR ;SAVE CURRENT BUF POINTER.  
RTS PC ;EXIT.  
98: MOV #MODQ,R2 ;MODULE TABLE ADDR TO R2.  
108: MOV (R2)+,R3 ;GET MODULE ADDR.  
BEQ 78 ;INVALID NAME, NO MODULE!  
MOV #AMODNM+1,R1

1863	011460	012704	000005		MOV	#5,R4	;WILL MATCH 5 CHARS.
1864	011464	122123		1181	CMPB	(R1)+,(3)+	;CHAR MATCH?
1865	011466	001370			BNE	108	;BR IF NOT.
1866	011470	005304			DEC	R4	;DONE?
1867	011472	001374			BNE	118	;BR IF NOT.
1868	011474	014237	001040		MOV	=(R2),MODPTR	;MODULE ADDR TO MODPTR.
1869	011500	010037	001036		MOV	R0,KBPTR	;SAVE CURRENT BUF POINTER.
1870	011504	062716	000004		ADD	#4,(SP)	;SET UP SUCCESS EXIT.
1871	011510	000207			RTS	PC	;EXIT.

1872							;OCTAL TO ASCII CONVERT ROUTINE.
1873							;CONVERTS ARGUMENT ON STACK TO ASCII, STORES STRING STARTING AT ADDRESS
1874							;GIVEN FOLLOWING THE CALL
1875							;EXTENDED ENTRY REQUIRES ADDITIONAL ARGUMENT ON STACK- AFTER
1876							;PUSHING LOWER 16 BITS ONTO STACK, PUSH EA BITS (SHIFTED TO POS 4,5)
1877							
1878	011512	005037	001112		OACNV:	CLR OASCEA	;CLEAR OUT UPPER BITS -- NOT USED
1879	011516	105037	001310		CLRB	ATE	;CLEAR 8 CHARACTER FLAG
1880	011522	000412			BR	CONVRT	;CONTINUE
1881							
1882	011524	105037	001310		OACNVX:	CLRB ATE	;CLEAR 8 CHARACTER FLAG
1883	011530	000403			BP	HOP	;CONTINUE
1884							
1885	011532	112737	000200	001310	OACNV8:	MOV8 #R17,ATE	;SET 8 CHARACTER FLAG
1886							
1887	011540	016637	000002	001112	HOP:	MOV 2(SP),OASCEA	
1888	011546	012816			MOV	(SP)+,(SP)	
1889	011550	004437	012244		CONVRT:	JSR R4,SAV04	;SAVE REGS 0-4.
1890	011554	016600	000014		MOV	12,(SP),R0	;GET OCTAL VALUE.
1891	011560	012501			MOV	(R5)+,R1	;GET DEST ADDR.
1892	011562	012702	000006		MOV	#6,R2	;SET CONVERT COUNT TO 6.
1893	011566	060201			ADD	R2,R1	;DEVELOP ADDR TO STORE 1ST CHAR.
1894	011570	105737	001310		TSTB	ATE	;SET UP TO DO 8 CHARACTERS ?
1895	011574	100002			BPL	38	;NO, CONTINUE
1896	011576	062701	000002		ADD	#2,R1	;YES, ADD 2 TO THE ADDRESS
1897	011602	010003		38:	MOV	R0,R3	;GET VALUE TO R3.
1898	011604	042703	177770		BIC	#177770,R3	;ISOLATE LEAST SIGNIFICANT DIGIT.
1899	011610	062703	000060		ADD	#60,R3	;CONVERT TO ASCII.
1900	011614	110341			MOV8	R3,=(R1)	;STORE IT.
1901	011616	042700	000007		BIC	#7,R0	;CLEAR DIGIT JUST CONVERTED.
1902	011622	006000			ROP	R0	;SHIFT IN NEXT DIGIT.
1903	011624	006000			ROR	R0	
1904	011626	006000			ROR	R0	
1905	011630	005302			DEC	R2	;DONE 6 DIGITS?
1906	011632	001363			BNE	38	;BR IF NOT.
1907	011634	105737	001310		TSTB	ATE	;SET UP TO DO 8 CHARACTERS ?
1908	011640	100410			BMI	46	;YES, DO IT
1909	011642	013702	001112		MOV	OASCEA,R2	
1910	011646	006202			ASR	R2	
1911	011650	006202			ASR	R2	
1912	011652	006202			ASR	R2	
1913	011654	006203			ADD	R2,R3	
1914	011656	110311			MOV8	R3,(R1)	
1915	011660	000425			BR	58	;GO FINISH UP
1916							
1917	011662	132737	000001	001310	48:	BITB #BIT0,ATE	;ALL DONE ?
1918	011670	001021			BNE	58	;YES, GET OUT
1919	011672	013700	001112		MOV	OASCEA,R0	;NO, LOAD UPPER NUMBERS TO BE CONVERTED
1920	011676	006300			ASL	R0	;LINE UP BITS BELONGING TO LOWER NUMBER
1921	011700	042700	177770		BIC	#177770,R0	;GIT RID OF ALL OTHERS
1922	011704	006003			ADD	R0,R3	;ADD IT IN TO COMPLETE THE NUMBER
1923	011706	110311			MOV8	R3,(R1)	;STORE IT
1924	011710	013700	001112		MOV	OASCEA,R0	;LOAD UPPER BITS AGAIN
1925	011714	006200			ASR	R0	;GIT RID OF THE 2 BITS
1926	011716	006200			ASR	R0	;JUST TAKEN CARE OF
1927	011720	012702	000002		MOV	#2,R2	;SET UP TO DO THE LAST 2 CHARACTERS

```

1928 011724 152737 000001 001310      B18B  #BIT0,ATE      ;SET 'ALL DONE' FLAG
1929 011732 000723                    BR      36          ;GO DO THE LAST 2 CHARACTERS
1930
1931 011734 004737 012256          58:    JSR      PC,RST04      ;RESTORE REGS. 0 THRU 4
1932 011740 012616                    MOV      (SP)+,(RP)      ;
1933 011742 000205                    RTS      R5            ;RETURN
    
```

```

1934                                     ;BINARY TO DECIMAL ASCII CONVERT ROUTINE.
1935                                     ;CONVERTS ARGUMENT ON STACK TO DECIMAL ASCII, STORES IT
1936                                     ;STARTING AT ADDRESS GIVEN AFTER THE CALL.
1937                                     ;REMOVES ARGUMENT FROM STACK BEFORE RETURNING
1938
1939 011744 004437 012244      BDCNV: JSR      R4,SAV04      ;SAVE REGS 0-4.
1940 011750 016601 000014      MOV      12,(SP),R1      ;GET BIN VALUE.
1941 011754 012700 016230      MOV      #DECVAL,P0      ;GET ADDR OF DECVAL STRING.
1942 011760 012702 012044      MOV      #TENPWR,R2      ;ADDR OF TENPWR TO R2.
1943 011764 012703 000005      MOV      #5,R3          ;SET UP TO DO 5 CONVERSIONS.
1944 011770 005004          18:    CLR      R4            ;CLEAR RESULT.
1945 011772 161201          28:    SUB      (R2),R1        ;SUBTRACT TEN POWER.
1946 011774 103402          BCS      36            ;BR IF UNSUCCESSFUL.
1947 011776 005204          INC      R4            ;ADD 1 TO RESULT.
1948 012000 000774          BP      28            ;DO IT AGAIN.
1949 012002 062201          38:    ADD      (R2)+,R1      ;RESTORE SUBTRACTED VALUE.
1950 012004 062704 000060      ADD      #60,R4          ;MAKE IT ASCII.
1951 012010 110420          MOVR    R4,(R0)+        ;SAVE IT.
1952 012012 005303          DEC      R3            ;DONE?
1953 012014 001365          BNE     18            ;BR IF NOT.
1954 012016 012501          MOV      (R5)+,R1      ;GET FINAL STORE ADDR.
1955 012020 012702 000005      MOV      #5,R2          ;GET DIGIT COUNT DESIRED.
1956 012024 060201          ADD      R2,R1          ;COMPUTE ADDR OF 1ST DIGIT.
1957 012026 114041          48:    MOVB   -(P0),-(R1)    ;TRANSFER CHAR.
1958 012030 005302          DEC      R2            ;DONE?
1959 012032 001375          BNE     48            ;BR IF NOT.
1960 012034 004737 012256      JSR      PC,RST04      ;RESTORE REGS 0-4.
1961 012040 012616          MOV      (SP)+,(SP)    ;
1962 012042 000205          RTS      R5            ;EXIT.
1963 012044 023420 001750 000144 TENPWR: 10000,,1000,,100,,10,,1
1964 012052 000012 000001
    
```

```

1965 ;RETURN TO KERNEL MODE
1966
1967 012056 042766 170000 000002 RTTK: BIC #170000,2(SP)
1968 012064 000002 RTI
1969
1970 ;
1971 ; ROUTINE TO CALC NEXT RANDOM NUMBER, USES A 29 BIT SHIFT REGISTER
1972 ; ALGORITHM, EXCLUSIVELY OR-ING THE 28TH AND 1ST(LOW ORDER) BITS
1973 ; INTO THE 29TH BIT AS IT SHIFTS RIGHT, SHIFTS 16 TIMES EACH TIME
1974 ; ITS CALLED. MODULES DO A RAND CALL THEN JUST REFERENCE RANNUM
1975 ; WHICH IS DEFINED AS GLOBAL IN DDXCOM
1976
1977 012066 004437 012244 RAND: JSP R4, SAV04 ;SAVE THE REGISTERS
1978 012072 013700 001162 MOV RANWRK,R0 ;GET UPPER BITS
1979 012076 013701 001160 MOV RANNUM,R1 ;GET LOWER ORDER 16 BITS
1980 012102 012702 000020 MOV #16.,R2 ;SHIFT 16 TIMES
1981 012106 006000 LOOP: ROR R0 ;SHIFT R0 INTO CARRY
1982 012110 006001 ROR R1 ;SHIFT CARRY INTO R1, R1 INTO CARRY
1983 012112 103410 BCS 16 ;FIRST HALF OF EX-OR PROCESS
1984
1985 012114 042700 010000 BIC #10000,R0 ;ASSUME OLD BIT 28 WAS ALSO CLEAR
1986 012120 032700 002000 BIT #2000,R0 ;CHECK THAT ASSUMPTION
1987 012124 001412 BEQ 28 ;BR IF RIGHT, ELSE
1988 012126 052700 010000 BIS #10000,R0 ;PUT BIT IN
1989 012132 000407 BR 28 ;GO SHIFT AGAIN
1990
1991 012134 052700 010000 16: BIS #10000,R0 ;ASSUME OLD BIT 28 WASN'T ALSO SET
1992 012140 032700 002000 BIT #2000,R0 ;CHECK THAT ASSUMPTION
1993 012144 001402 BEQ 28 ;BR IF RIGHT
1994 012146 042700 010000 BIC #10000,R0 ;ELSE 2 1'S = A ZERO BIT
1995
1996 012152 005302 28: DEC R2 ;DONE 16 TIMES?
1997 012154 001354 RLOOP BNE RLOOP ;BR IF NO
1998 012156 010037 001162 MOV R0,RANWRK ;SAVE CURRENT VALUES
1999 012162 010137 001160 MOV R1,RANNUM
2000 012166 004737 012256 JSR PC, PST04 ;RESTORE REG'S
2001 012172 000002 RTI ;RETURN FROM TRAP CALL
    
```

```

2002 ;SUBROUTINE TO GET PAGE ADDRESS FIELD VALUE (TOP 12 BITS OF ADDRESS)
2003 ;CALL VIA:
2004 ; JSR R5, TOP12S
2005 ; ADDRESS OF TABLE CONTAINING LOWER 16 BITS, TOP 2 BITS (SHIFTED),
2006 ; AND SLOT FOR RETURNED VALUE OF ADDRESS
2007
2008 012174 010046 TOP12S: MOV R0,=(SP) ;SAVE THE OLD VALUE OF R0
2009 012176 012500 MOV (R5)+,R0 ;GET THE ADDRESS OF THE TABLE-PUT IT IN R0
2010 012200 017046 MOV (R0)+,=(SP) ;FETCH THE ADDRESS TO BE ACTED ON TO THE STACK
2011 012202 006216 ASR (SP) ;ELIMINATE THE SIX LOW ORDER BITS, THIS ONE DROPS BIT0
2012 012204 006216 ASR (SP) ;THIS ROTATION DROPS BIT1
2013 012206 006216 ASR (SP) ;THIS ROTATION DROPS BIT2
2014 012210 006216 ASR (SP) ;THIS ROTATION DROPS BIT3
2015 012212 006216 ASR (SP) ;THIS ROTATION DROPS BIT 4
2016 012214 006216 ASR (SP) ;AND THIS ROTATION DROPS BITS
2017 012216 042716 176000 BIC #176000,(SP) ;ISOLATE THE 12 HIGH ORDER BITS IN THE LOWER
2018 012222 011660 000002 MOV (SP),2(R0) ;SAVE THE 12 HIGH ORDER BITS IN THE TABLE'S 3RD ENTRY
2019 012226 012016 MOV (R0)+,(SP) ;FETCH THE EXTENDED ADDRESS BITS FROM THE 2ND ENTRY
2020 012230 000316 SWAB (SP) ;MOVE THE BITS TO BIT POSITIONS 6+7, FIRST, TO 8+9,
2021 012232 006216 ASR (SP) ;THEN MOVE THEM TO 7 + 8
2022 012234 006216 ASR (SP) ;FINALLY ROTATE THEM TO 6 + 7
2023 012236 052610 RIS (SP)+,(R0) ;ADD THE EXTENDED ADDRESS BITS TO THE 3RD ENTRY
2024 012240 012600 MOV (SP)+,R0 ;RESTORE THE OLD VALUE OF R0
2025 012242 000205 RTS R5 ;RETURN TO THE CALLING PROCEDURE
2026
    
```

```

2027                                     ;SAVE REGS 0-4 SUBROUTINE, (CALL VIA JSR R4,SAVQ4)
2028
2029 012244 010346                       SAVQ4: MOV   R3,-(SP)           ;SAVE R3
2030 012246 010246                       MOV   R2,-(SP)           ;SAVE R2
2031 012250 010146                       MOV   R1,-(SP)           ;SAVE R1
2032 012252 010046                       MOV   R0,-(SP)           ;SAVE R0
2033 012254 010407                       MOV   R4,PC              ;R4 IS ALREADY SAVED
2034
2035                                     ;RESTORE REGS 0-4 SUBROUTINE, (CALL VIA JSR PC,RST04)
2036
2037 012256 012604                       RST04: MOV   (SP)+,R4      ;RETURN ADDRESS
2038 012260 012600                       MOV   (SP)+,R0          ;RESTORE R0
2039 012262 012601                       MOV   (SP)+,R1          ;R1
2040 012264 012602                       MOV   (SP)+,R2          ;R2
2041 012266 012603                       MOV   (SP)+,R3          ;R3
2042 012270 000204                       RTS    R4                ;RESTORE R4 AND RETURN
    
```

```

2043                                     ;CLEAR VARIABLES AND QUEUES, AND LOAD TRAP CATCHER STARTING AT 70
2044
2045 012272 012700 001006                 CLRQUS: MOV   #IOQUE,R0   ;CLEAR VARIABLES.
2046 012276 012701 000140                 MOV   #TKS-IOQUE,R1
2047 012302 105020                       10:   CLRB   (R0)+
2048 012304 005301                       DEC   R1
2049 012306 001375                       BNE   18
2050 012310 012700 016606                 MOV   #IOQ,R0           ;INITIALIZE QUEUE POINTERS
2051 012314 010037 001074                 MOV   R0,IOQ1
2052 012320 010037 001076                 MOV   R0,IOQ2
2053 012324 012737 017116 001100         MOV   #TYPEQ,TYPQ1
2054 012332 012737 017116 001102         MOV   #TYPEQ,TYPQ2
2055 012340 012701 000764                 MOV   #IOQL+TYPQL,R1   ;SET UP TO CLEAR QUEUES.
2056 012344 105020                       20:   CLRB   (R0)+
2057 012346 005301                       DEC   R1                ;DONE?
2058 012350 001375                       BNE   28                ;RR IF NOT.
2059 012352 000207                       RTS    PC
    
```

```

2060 ;LOAD TRAPCATCHER IN VECTOR AREA
2061
2062 012354 012700 000072 CLRVEC: MOV #72,R0 ;FILL VECTOR AREA WITH .+2
2063 012360 012701 000070 MOV #70,R1 ;AND HALT.
2064 012364 010021 1$! MOV R0,(R1)+
2065 012366 005021 CLR (R1)+
2066 012370 010100 MOV R1,R0
2067 012372 005720 TST (R0)+
2068 012374 020027 001002 CMP R0,#1002 ;FILLED UP?
2069 012400 001371 BNE 1$ ;BR IF NOT.
2070 012402 000207 RTS PC ;YES. EXIT.
2071
2072 ;TRAP INTERPRETER ROUTINE.
2073
2074 012404 010046 TRPINT: MOV R0,=(SP) ;PUSH R0.
2075 012406 016600 000002 MOV 2(SP),R0 ;GET TRAP PC.
2076 012412 014000 MOV -(R0),R0 ;GET TRAP CALL.
2077 012414 006300 ASL R0 ;MULTIPLY BY 2.
2078 012416 082700 ADD #TRPTAB-TRP2,R0 ;FORM TABLE ADDRESS
2079 012422 020027 012516 CMP R0,#TRPLIM ;WITHIN LIMITS?
2080 012426 103002 BHTS 1$ ;BR IF NOT.
2081 012430 011000 MOV #R0,R0 ;GET ROUTINE ADDRESS
2082 012432 000200 RTS R0 ;GO TO ROUTINE, RESTORE R0.
2083 012434 104406 014223 1$! MSG$,INVTYP ;ERROR INVALID TRAP CALL. CRASH SYSTEM.
2084 012440 104406 014620 MSG$,HLT ;"HALT"
2085 012444 000000 HALT
2086 011000 TRP2=11000
2087 000000 TRAPX=0
2088
2089 012446 TRPTAB:
2090 012446 TRPDEF EXITS,EXIT.
2091 012450 TRPDEF QUES,QUE. ;POINTER FOR TRAP CALL EXITS
2092 012450 TRPDEF QUES,QUE. ;POINTER FOR TRAP CALL QUES
2093 012452 TRPDEF ENDP$,TYPQ. ;POINTER FOR TRAP CALL ENDP$
2094 012452 TRPDEF ENDS,TYPQ. ;POINTER FOR TRAP CALL ENDS
2095 012454 TRPDEF ERRORS,TYPQ2. ;POINTER FOR TRAP CALL ERRORS
2096 012456 TRPDEF DATER$,TYPQ2. ;POINTER FOR TRAP CALL DATER$
2097 012456 TRPDEF MSG$,TYPQ1. ;POINTER FOR TRAP CALL MSG$
2098 012456 TRPDEF BREAK$,LDBRK. ;POINTER FOR TRAP CALL BREAK$
2099 012460 TRPDEF ERRN$,TYPQ2. ;POINTER FOR TRAP CALL ERRN$
2100 012460 TRPDEF MSGN$,TYPQ. ;POINTER FOR TRAP CALL MSGN$
2101 012462 TRPDEF GWRBUF$,GWRBUF. ;POINTER FOR TRAP CALL GWRBUF$
2102 012462 TRPDEF GETPA$,GETPA. ;POINTER FOR TRAP CALL GETPA$
2103 012464 TRPDEF COATA$,CDATA. ;POINTER FOR TRAP CALL CDATA
2104 012464 TRPDEF MAP22$,MAP22.
2105 012466
2106 012466
2107 012470
2108 012472 005324
2109 012472 005324
2110 012472 013646
2111 012474
2112 012474 013770
2113 012476
2114 012476 006626
2115 012500
    
```

```

2116 012500 014012 MAP22. ;POINTER FOR TRAP CALL MAP22$
2117 012502 TRPDEF MSGS$,TYPQ. ;POINTER FOR TRAP CALL MSGS$
2118 012502 005324 TYPQ. ;POINTER FOR TRAP CALL DATCK$
2119 012504 TRPDEF DATCK$,DATCK. ;POINTER FOR TRAP CALL DATCK$
2120 012504 006616 DATCK. ;POINTER FOR TRAP CALL RAND$
2121 012506 TRPDEF RAND$,RAND. ;POINTER FOR TRAP CALL DERRX$
2122 012506 012066 RAND. ;POINTER FOR TRAP CALL DERRX$
2123 012510 TRPDEF DERRX$,TYPQ3. ;POINTER FOR TRAP CALL MSGD$
2124 012510 005150 TYPQ3. ;POINTER FOR TRAP CALL MSGD$
2125 012512 TRPDEF MSGD$,TYPQ4. ;POINTER FOR TRAP CALL RTTK$
2126 012512 005102 TYPQ4. ;POINTER FOR TRAP CALL RTTK$
2127 012514 TRPDEF RTTK$,RTTK. ;POINTER FOR TRAP CALL
2128 012514 012056 RTTK. ;POINTER FOR TRAP CALL
2129 012516 TRPLIM:
    
```

```

2130 ;BUS ERROR AND RESERVED INSTRUCTION TRAP ROUTINES
2131 ;CHECK MODE BITS OF PSW ON STACK (MAYBE USER IF KT11 IN USE)
2132 ;ORDER OF ARGUMENTS: VIRTUAL PC ON STACK, PSW ON STACK, STACK POINTER
2133 ;AFTER TRAP, VIRTUAL ADDRESS TRAPPED TO
2134
2135 012516 012746 000004 BUSERR: MOV #4,-(SP) ;INDICATE BUS ERROR TRAP,
2136 012822 000402 BR RESIA
2137 012824 012746 000010 RESINT: MOV #10,-(SP) ;INDICATE RESERVED INSTRUCTION TRAP,
2138 012830 000240 RESIA: NOP
2139 012832 010605 MOV SP,R5 ;SAVE THE STACK POINTER,
2140 012834 012706 016606 MOV #SPSPEC,SP ;PUT INFORMATION INTO SPECIAL MONITOR STACK
2141 012840 012546 MOV (R5),-(SP) ;PUSH TRAP VECTOR ON STACK
2142 012842 010546 MOV R5,-(SP) ;PUSH STACK POINTER ON STACK
2143 012844 016546 000002 MOV 2(R5),-(SP) ;PUSH THE PSW ON STACK
2144 012850 011546 MOV (R5),-(SP) ;PUSH THE PC ON STACK
2145 012852 012706 016546 MOV #SPBOT,SP ;RESTORE MONITOR STACK POINTER
2146 012856 004737 006230 JSR PC,CTRLX ;CLEAR QUEUES, CLEAN UP, MAP TO BANKS 0-1
2147 012862 105237 001244 INCB SYSCNT ;COUNT A SYSTEM ERROR
2148 012866 012706 016576 MOV #SPSPEC-10,SP ;SETUP SPECIAL STACK POINTER
2149 012872 005037 001060 CLR IOBKID ;ALLOW TYPING
2150 012876 112737 000001 001013 MOV# #1,BRAKE ;PREVENT MODULE STARTUP DURING MESSAGE
2151 012804 104406 014512 MSG#,SYSERR ;TYPE SYS ERROR FAILURE,
2152 012610 012705 011000 MOV #ITOA,R5
2153 012614 004715 JSR PC,(R5) ;TYPE ERR PC,
2154 012616 004715 JSR PC,(R5) ;TYPE ERR PSW,
2155 012620 004715 JSR PC,(R5) ;TYPE SP AFTER ERROR
2156 012622 004715 JSR PC,(R5) ;TYPE TRAP VECTOR ADDRESS
2157
2158 012624 012706 016546 MOV #SPROT,SP ;RESTORE MONITOR STACK POINTER
2159 012630 104406 014122 28: MSG#,DOT ;TYPE CRLF AND DOT
2160 012634 000426 BR PWRUA ;ATTEMPT TO CONTINUE

```

```

2161 ;POWER DOWN AND POWER UP ROUTINES,
2162
2163 012636 PWRDN: 18:
2164 012636 012737 012646 000024 MOV #PWRUP,PWRFV ;SET UP POWER UP VECTOR,
2165 012644 000000 HALT ;POWER DOWN HALT
2166 012646 012737 012636 000024 PWRUP: MOV #PWRDN,PWRFV ;SET UP POWER DOWN VECTOR,
2167 012654 012706 016546 MOV #SPBOT,P6 ;RESET STACK,
2168 012660 005237 001246 INC PWRCNT ;COUNT A POWER FAIL
2169 012664 105037 001274 CLR# RUNRFL ;CLEARRESUME RUN FLAG
2170 012670 004737 006230 JSR PC,CTRLX ;CLEAR QUEUES AND INITIALIZE KT11 IF IN USE,
2171 012674 005037 001060 CLR IOBKID ;ALLOW TYPING
2172 012700 112737 000001 001013 MOV# #1,BRAKE ;PREVENT MODULE STARTUP
2173 012706 104406 014554 MSG#,PWRFAI ;TYPE POWER FAILURE MESSAGE,
2174 012712 PWRUA:
2175 012712 105737 001257 36: TSTB RMODE ;WERE WE IN RUN MODE?
2176 012716 001021 BNE LOGICA ;BR IF YES TO RESTART,
2177 012720 000400 BR CHNOUT ;NO, CONTINUE

```

```

2178 ;ROUTINE TO EXIT TO CHAIN MONITOR, OR RETURN TO KYBD RTN.
2179 ;PROGRAM SHOULD ALWAYS BE IN BANK 0 BY HERE
2180
2181 012722 105737 001260 CHNOUT: TSTR CHN ;IN CHAIN MODE?
2182 012726 100402 BMI 18 ;BR IF YES,
2183 012730 000137 007364 JMP COMCON ;BACK TO KYBD SERVICE.
2184 012734 105737 001252 10: TSTB KIPRES
2185 012740 001402 BFC 28 ;BRANCH IF NO KT11
2186 012742 005037 177572 CLR #SSRO ;TURN OFF KT11
2187 012746 013700 000042 28: MOV 42,R0
2188 012752 004710 LOGIC: JSR PC,(R0) ;RETURN TO MONITOR.
2189 012754 000240 000240 000240 ,WORD NOP,NOP,NOP
2190 012762 LOGICA:
2191 012762 105737 001260 50: TSTB CHN
2192 012766 001402 BEQ 66
2193 012770 000137 001414 JMP START
2194 012774 000137 007476 60: JMP RUNRES
    
```

```

2195 ;SUBROUTINE TO ROTATE WRITE BUFFERS IF ROTI INDICATOR =1.
2196
2197 013000 105737 001261 ROTBUF: TSTR ROTI ;ROTATION ALLOWED?
2198 013004 001525 BEQ ROTXT ;BR IF NOT.
2199 013006 105737 001262 TSTR ROTL ;IS ROTATION LOCKED ?
2200 013012 100016 RPL 18 ;NO, CONTINUE
2201 013014 105737 001275 TSTB LOCK ;YES, IS THE PROGRAM LOCKED ?
2202 013020 001013 BNE 18 ;YES, DON'T SET STOP
2203 013022 005337 001264 DEC ROTCNT ;NO, TIME TO RELOCATE ?
2204 013026 003114 BGT ROTXT ;NO, CONTINUE
2205 013030 105737 001320 TSTB WSTOP ;ALLOW STOP TO SET ?
2206 013034 001002 BNE 108 ;NO, CONTINUE
2207 013036 105237 001024 INCB STOP ;YES, SET THE STOP FLAG
2208 013042 013737 001266 001264 108: MOV ROTNUM,ROTCNT ;RESTORE THE COUNTER
2209 013050 105737 001024 10: TSTB STOP ;IF STOP IS SET, DON'T ROTATE
2210 013054 001101 BNE ROTXT
2211 013056 105737 001262 TSTB ROTL ;ROTATE LOCKED?
2212 013062 100476 BMI ROTXT ;BR IF YES
2213 013064 105737 001021 TSTB MDXCTR ;IF COUNTER NOT INITIALIZED, DON'T ROTATE
2214 013070 100473 BMI ROTXT
2215 013072 013746 001212 MOV WBUF,=(SP) ;CHECK CURRENT VALUE
2216 013076 042716 160000 BIC #160000,(SP)
2217 013102 022726 016000 CMP #16000,(SP)+
2218 013106 003016 BGT 38
2219 013110 105737 001021 TSTB MDXCTR ;IF NOT LAST IN BANK, BRANCH
2220 013114 001403 BEQ 28 ;HAVE ALL MODULES USED A VALUE IN THIS BANK?
2221 013116 105737 001302 DECB #GWBFC ;YES- TIME TO CHANGE BANKS
2222 013122 001056 BNE ROTXT ;NO, DEC TIMEOUT COUNTER
2223 013124 113737 001022 001021 28: MOVW MDXCNT,MDXCTR ;EXIT UNLESS TIMEOUT HAS OCCURRED
2224 013132 113737 001031 001302 MOVW ROTCT,#GWBFC ;RESET COUNTERS
2225 013140 105137 001023 COMR WBF12,DFFBSET ;TOGGLE INDICATOR
2226 013144 023737 001216 001222 38: CMP WBUF12,DFFBSET ;LOCORE ROTATION?
2227 013152 062737 000020 001216 ADD #20,WBUF12
2228 013160 023737 001216 001206 CMP WBUF12,HCOR12 ;CHECK FOR TOP OF WRITE BUFFER AREA
2229 013166 103417 BLO ROTB2 ;IF NOT, BRANCH
2230 013170 105737 001263 ROTB1: TSTB RELOC ;RELOCATION ALLOWED?
2231 013174 001426 BEQ ROTB3 ;NO- BRANCH
2232 013176 105737 001275 TSTB LOCK ;LOCKED?
2233 013202 001023 BNE ROTB3 ;YES- BRANCH
2234 013204 105737 001320 TSTB WSTOP ;ALLOW STOP TO SET ?
2235 013210 001002 BNE 18 ;NO, CONTINUE
2236 013212 105237 001024 INCB STOP ;YES, SET STOP
2237 013216 162737 000020 001216 18: SUB #20,WBUF12 ;RESTORE WBUF
2238 013224 000415 BR ROTXT ;EXIT
2239 013226 062737 002000 001212 ROTB2: ADD #2000,WBUF ;BUMP ADDR- POINT 512 WORDS HIGHER
2240 013234 103003 BCC 18
2241 013236 062737 000020 001214 18: ADD #20,WBFEA
2242 013244 004737 010166 18: JSR PC,SETWB1 ;CALCULATE MXWBF
2243 013250 000403 BR ROTXT
2244 013252 004737 010126 ROTB3: JSR PC,SETWBF
2245 013256 000400 BR ROTXT ;RETURN
2246
2247 013260 000207 ROTXT: RTS PC ;RETURN
    
```

```

2248
2249
2250 ;SUBROUTINE TO CHECK IF BUFFER ROTATION ENABLED, AND TO TYPE
2251 ;THE ROTATION RANGE.
2252 013262 105737 001303 ROTCHK: TSTB MEMERF ;ANY MEMORY ERRORS?
2253 013266 001406 BEQ 16 ;NO- BRANCH
2254 013270 104406 016061 MSGS,HERR1
2255 013274 104406 014976 MSGS,MEMCHG
2256 013300 105037 001303 CLR B MEMERF
2257 013304 105737 001261 18: TSTB ROTI ;ROTATE ENABLED?
2258 013310 001444 BEQ 28 ;BR IF NOT.
2259 013312 104406 014630 MSGS,ROTENB
2260 013316 013746 001200 MOV LOCORE,-(SP) ;TYPE LOWER LIMIT.
2261 013322 004737 011000 JSR PC,ITOA
2262 013326 005037 001120 CLR SAVHI ;ZERO ANY HIGH ORDER BITS
2263 013332 013737 001206 MOV HCDP12,SAVLO ;COPY INTO LOW ORDER BITS
2264 013340 006337 001116 ASL SAVLO ;MAKE IT A 16-BIT NUMBER
2265 013344 006337 001116 ASL SAVLO
2266 013350 006337 001116 ASL SAVLO
2267 013354 006337 001116 ASL SAVLO
2268 013360 006337 001116 ASL SAVLO
2269 013364 006337 001116 ASL SAVLO
2270 013370 162737 000002 001116 SUB #2,SAVLO
2271 013376 103003 BCC 38
2272 013400 162737 000020 001120 SUB #20,SAVHI
2273 013406 013746 001116 38: MOV SAVLO,-(SP) ;TYPE UPPER LIMIT
2274 013412 013746 001120 MOV SAVHI,-(SP)
2275 013416 004737 010710 JSR PC,ITOA
2276
2277 013422 000207 28: RTS PC ;EXIT.

```

```

2278 ; THIS ROUTINE WILL CONVERT THE GIVEN BINARY
2279 ; NUMBER INTO HOURS, MINUTES, AND SECONDS. LOCATIONS
2280 ; HOLDING THE FINAL NUMBERS WILL BE TAGGED AS: HOURS,
2281 ; MINITS, AND SEKND5. HOURS MUST BE ON AN ODD BOUNDARY.
2282 ; MINITS AND SEKND5 MUST BE ON EVEN BOUNDARIES.
2283 ; TEMPORARY LOCATION HOLD5 MUST BE ON AN ODD BOUNDARY.
2284 ;
2285 ; CALL VIA: JSR PC,HMS
2286 ;
2287 ; ADR ;ADDRESS OF THE TIME
2288 ; ADRX ;ADDRESS OF EXTENDED TIME BITS
2289 013424 004437 012244 HMS: JSR R4,SAV04 ;SAVE REGS. 0-4
2290 013430 016600 000012 MOV 10,(SP),R0 ;GET ADP OF OF ADR TIME
2291 013436 005723 MOV R0,R3 ;COPY IT INTO REG. 3
2292 013440 017000 000000 TST (R3)+ ;GET ADR OF EXTENDED TIME BITS ADR
2293 013444 017303 000000 MOV @R0,R0 ;PUT THE TIME IN REG. 0
2294 013450 005001 MOV @R3,R3 ;PUT EXTENDED TIME BITS INTO REG. 3
2295 013452 005002 CLR R1 ;ZERO MINUTES COUNTER
2296 CLR R2 ;ZERO HOURS COUNTER
2297 013454 022700 007020 18: CMP #3600,,R0 ;LESS THAN 1 HOUR?
2298 013460 101013 BHI 38 ;YES, GO LOOK FOR MORE HOURS
2299 013462 162700 007020 SUB #3600,,R0 ;NO, SUBTRACT 1 HOUR'S WORTH OF SECONDS
2300 013466 005202 INC R2 ;COUNT AN HOUR
2301 013470 000771 BR 18 ;CHECK FOR MORE HOURS
2302
2303 013472 022700 000074 28: CMP #60,,R0 ;LESS THAN 1 MINUTE?
2304 013476 101014 BHI 48 ;YES, ALL DONE
2305 013800 162700 000074 SUB #60,,R0 ;NO, SUBTRACT 1 MINUTE'S WORTH OF SECONDS
2306 013804 005201 INC R1 ;COUNT A MINUTE
2307 013806 000771 BR 28 ;CHECK FOR MORE MINUTES
2308
2309 013810 005703 38: TST R3 ;ANY EXTENDED TIME BITS ?
2310 013812 001767 BEQ 24 ;NO, GO LOOK FOR MINUTES
2311 013814 062702 000021 ADD #17,,R2 ;YES, ADD IN 17. HOURS
2312 013820 062700 010360 ADD #4336,,R0 ;ADD IN 1 HR 12 MIN 16 SEC
2313 013824 005303 DEC R3 ;GET RID OF ONE TIME EXTENDED BIT
2314 013826 000752 BR 18 ;CONTINUE
2315
2316 013830 010246 48: MOV R2,-(SP) ;PUT THE HOURS ON THE STACK
2317 013832 004537 011744 JSR R5,BDCNV ;CONVERT IT TO ASCII
2318 013836 015507 HOLD5 ;AND PUT IT HERE
2319 013840 113737 015511 001353 MOV B HOLD5+2,HOURS ;LOAD HOURS
2320 013846 013737 015512 001354 MOV B HOLD5+3,HOURS+1 ;LOAD HOURS
2321 013854 010146 MOV R1,-(SP) ;PUT THE MINUTES ON THE STACK
2322 013856 004537 011744 JSR R5,BDCNV ;CONVERT IT TO ASCII
2323 013862 015507 HOLD5 ;AND PUT IT HERE
2324 013864 013737 015512 001356 MOV B HOLD5+3,MINITS ;LOAD MINUTES
2325 013872 010046 MOV R0,-(SP) ;PUT THE SECONDS ON THE STACK
2326 013874 004537 011744 JSR R5,BDCNV ;CONVERT IT TO ASCII
2327 013880 015507 HOLD5 ;AND PUT IT HERE
2328 013882 013737 015512 001360 MOV B HOLD5+3,SEKND5 ;LOAD SECONDS
2329
2330 013610 004737 012256 JSR PC,RST04 ;RESTORE REGS 0-4
2331 013614 062716 000004 ADD #4,(SP) ;SET PROPER RETURN PC
2332 013620 000207 RTS PC ;AND RETURN
2333

```

```

2334
2335
2336
2337
2338 013622 013706 001136
2339 013626 008046
2340 013630 012746 013636
2341 013634 000002
2342 013636 005037 001374
2343 013642 000137 004550
    ; THIS IS THE CALLING SEQUENCE FOR THE CKHUNG ROUTINE
HUNGCK: MOV     SPSAV,SP      ;RESTORE MONITOR STACK POINTER
          CLR     -(SP)       ;CLEAR PSW ON STACK
          MOV     #18,-(SP)   ;SETUP RETURN PC
          RTI     ;LOAD NEW PSW AND CONTINUE
          CLR     CSTART     ;INDICATE NO MODULES DOING EOP
          JMP     ENDSVG     ;GO CHECK FOR OTHER MODULES RUNNING
    
```

```

2344
2345
2346 013646 010146
2347 013650 010246
2348 013652 017601 000094
2349 013656 010102
2350 013660 062701 000074
2351 013664 013721 001212
2352 013670 013721 001214
2353 013674 012111
2354 013676 021137 001236
2355 013702 101402
2356 013704 013711 001236
2357 013710 108737 001021
2358 013714 100410
2359 013716 126237 000005 001023
2360 013724 001404
2361 013726 105162 000005
2362 013732 105337 001021
2363 013736 012602
2364 013740 012601
2365 013742 052716 000002
2366 013746 105337 001035
2367 013752 100005
2368 013754 113737 001305 001035
2369 013762 004737 013000
2370 013766 000002
2371
2372
2373
2374 013770 010146
2375 013772 017601 000002
2376 013776 012121
2377 014000 005011
2378 014002 012601
2379 014004 062716 000002
2380 014010 000002
2381
2382
    ;GET WRITE BUFFER ADDRESS SERVICE
GWBUF,i MOV     R1,-(SP)      ;SAVE REGISTER
          MOV     R2,-(SP)
          MOV     #4(SP),R1   ;GET ADDRESS OF MODULE
          MOV     R1,R2
          ADD     #WBUFPA,R1
          MOV     WBUF,(R1)+  ;LOAD ADDRESS OF WRITE BUFFER
          MOV     WRFEA,(R1)+ ;LOAD EXTENDED ADDRESS BITS
          MOV     (R1)+,(R1)  ;MOVE SIZE REQUESTED INTO SIZE ALLOWED
          CMP     (R1),#XWBF  ;REQUESTED SIZE OK?
          BLOS   1$          ;YES- BRANCH
          MOV     #XWBF,(R1)  ;NO- SET ALLOWED SIZE
          TSTB   MDXCTR      ;IF NOT YET SETUP, SKIP
          BVI   2$
          CMPR   XFLAG(2),WBFLG ;IF SAME, HAS ALREADY USED THIS BANK
          BEQ   2$
          COMB   XFLAG(2)    ;IF NOT, SET FLAG TO SAME
          DECB   MDXCTR      ;AND COUNT DOWN MODULES
          MOV     (SP)+,R2    ;RESTORE REGISTERS
          MOV     (SP)+,R1
          ADD     #2,(SP)     ;SETUP RETURN PC
          DECB   ROT
          BPL   3$
          MOVB   ROTN,ROT    ;GO CHANGE WRITE BUFFER ADDRESS
          JSR   PC,ROTBUF
          RTI
          ;RETURN
    
```

```

2383 ;THIS ROUTINE ALLOWS RM70 DEVICES TO BE RUN WITH THE STANDARD MONITOR,
2384
2385
2386 014012 010146 MAP22: MOV R1, -(SP) ;SAVE REG. 1
2387 014014 017601 000002 MOV #2(SP), R1 ;GET THE 1ST ADDRESS OF THE TABLE
2388 014020 012161 000002 MOV (R1)+, 2(R1) ;GIVE BACK THE SAME 18 BITS
2389 014024 011161 000004 MOV (R1), 4(R1) ;THAT WERE GIVEN TO BE CONVERTED
2390 014030 006261 000004 ASR 4(R1) ;SHIFT BACK FROM POSITIONS 4 AND 5
2391 014034 006261 000004 ASR 4(R1) ;TO POSITIONS 0 AND 1
2392 014040 006261 000004 ASR 4(R1) ;
2393 014044 006261 000004 ASR 4(R1) ;
2394 014050 012601 MOV (SP)+, R1 ;RESTORE REG. 1
2395 014052 062716 000002 ADD #2, (SP) ;SKIP OVER VA TO GET THE RETURN PC
2396 014056 000002 RTI ;RETURN
2397 014060 000207 MVCODE: RTS PC ;DO NOTHING

```

```

2398 ;PURE MESSAGES
2399
2400 014062 042045 041505 054057 TITLE: .ASCIZ ' *DFC/X11, DXQAE-D SHORT MONITOR'
2401 014070 030461 020056 054104
2402 014076 040521 026505 020104
2403 014104 044123 051117 020124
2404 014112 047515 044516 047524
2405 014120 000122
2406 014122 027045 000 DOT: .ASCIZ '%,'
2407 014125 045 000 CR: .ASCIZ '%*'
2408 014127 045 000045 CRCR: .ASCIZ '%**'
2409 014132 040440 000124 AAT: .ASCIZ '% AT'
2410 014136 051045 047125 051440 SUMMARY: .ASCIZ '%RUN SUMMARY'
2411 014144 046525 040515 054522
2412 014152 000
2413 014155
2414 014153 136 022503 000 CTRLC: .ASCIZ '%C%'
2415 014157 045 020045 052521 QDVRFL: .ASCIZ '%** QUEUE OVERFLOW --- SYSTEM CRASH%'
2416 014164 052505 020105 053117
2417 014172 051105 046106 053517
2418 014200 026440 026455 051440
2419 014206 051531 042524 020115
2420 014214 051103 051501 022510
2421 014222 000
2422 014223 045 020045 047111 INVTRP: .ASCIZ '%** INVALID TRAP CALL --- SYSTEM CRASH%'
2423 014230 040526 044514 020104
2424 014236 051124 050101 041440
2425 014244 046101 020114 026455
2426 014252 020055 054523 052123
2427 014260 046505 041440 040522
2428 014266 044123 000045
2429 014272 041045 042101 041440 INVCMD: .ASCIZ '%BAD CMD%'
2430 014300 042115 000045
2431 014304 041445 052116 020114 INVCD1: .ASCIZ '%CNTL C ONLY%'
2432 014312 020103 047117 054514
2433 014320 000045
2434 014322 047045 020117 052113 INVCD2: .ASCIZ '%NO K711%'
2435 014330 030461 000045
2436 014334 047045 020117 052113 INVCD3: .ASCIZ '%NO K111%'
2437 014342 030461 000045
2438 014346 047045 020117 047515 INVCD4: .ASCIZ '%NO WDS SEL%'
2439 014354 051504 051440 046105
2440 014362 000045
2441 014364 041045 042101 040440 INVADR: .ASCIZ '%BAD ADR/DATA%'
2442 014372 051104 042057 052101
2443 014400 022501 000
2444 014403 045 040502 020104 INVNAM: .ASCIZ '%BAD NAME%'
2445 014410 040516 042515 000045
2446 014416 045445 052502 020106 KBOFLO: .ASCIZ '%KBUF OFLO'
2447 014424 043117 047514 000
2448 014431 045 052045 020117 CLR40: .ASCIZ '%TO EXERCISE LOAD MEDIUM YOU MUST CLEAR LOC 40%'
2449 014436 054105 051105 044503
2450 014444 042523 046040 040517
2451 014452 020104 042515 044504
2452 014460 046525 054440 052517
2453 014466 046440 051525 020124

```

2454	014474	046103	040505	020122				
2455	014502	047514	020103	030064				
2456	014510	000045						
2457	014512	051445	051531	042440	SYSERRi	,ASCIZ	"%SYS ERR (VIRT,PC,PSW,SP,VECT): "	
2458	014520	051122	024040	044526				
2459	014526	052122	050056	026103				
2460	014534	051520	026127	050123				
2461	014542	053054	041505	024524				
2462	014550	020072	000040					
2463	014554	022445	047520	042527	PWRFAIi	,ASCIZ	"%%POWER FAILURE%%"	
2464	014562	020122	040506	046111				
2465	014570	051125	022505	000045				
2466	014576	051045	047125	046440	MEMCHGi	,ASCIZ	"%RUN MEMORY TESTS"	
2467	014604	046505	051117	020131				
2468	014612	042524	052123	000123				
2469	014620	044045	046101	022524	HLTi	,ASCIZ	"%HALT%%"	
2470	014626	000045						

2471	014630	053445	044522	042524	ROTEENBi	,ASCIZ	"%WRITE BUFFER ROTATION ENABLED. RANGE: "	
2472	014636	041040	043125	042506				
2473	014644	020122	047522	040524				
2474	014652	044524	047117	042440				
2475	014660	040516	046102	042105				
2476	014666	020056	040522	043516				
2477	014674	035105	000040					
2478								
2479	014700	053445	044522	042524	ROTDISi	,ASCIZ	"%WRITE BUFFER ROTATION DISABLED"	
2480	014706	041040	043125	042506				
2481	014714	020122	047522	040524				
2482	014722	044524	047117	042040				
2483	014730	051511	041101	042514				
2484	014736	000104						
2485	014740	045445	030524	020061	KTONMi	,ASCIZ	"%KT11 ENABLED"	
2486	014746	047105	041101	042514				
2487	014754	000104						
2488	014756	051445	044527	041524	NOSWRi	,ASCIZ	"%SWITCH REGISTER AT LOC. 1004%"	
2489	014764	020110	042522	044507				
2490	014772	052123	051105	040440				
2491	015000	020124	047514	027103				
2492	015006	030440	030060	022464				
2493	015014	000						
2494		015016					.EVEN	

```

2495 ;ROUTINE TO LOAD WORST CASE UNIBUS PATTERN IN ALL FREE MEMORY
2496
2497 015016 004437 012244 LOADWC: JSR R4,SAY04
2498 015022 013702 001204 MOV LCDR12,R2
2499 015026 063702 001222 ADD OFFSET,R2 ;SETUP 12 BIT STARTING ADDRESS
2500 015032 023702 001206 CMP HCOR12,R2
2501 015036 001425 BEQ B6 ;ANY MEMORY TO LOAD ?
2502 015040 013700 001200 MOV LOCORE,R0 ;NO, GET OUT
2503 015044 012704 015132 28: MOV #WCASE,R4 ;IF NO KT11, R0 WILL CONTAIN PHYSICAL ADDRESS
2504 015050 105737 001252 38: KTPRES ;LOAD ADDRESS OF WORST CASE DATA TABLE
2505 015054 001404 ;USE KT11?
2506 015056 012700 040000 BEQ B5 ;NO= BRANCH
2507 015062 010237 172344 48: MOV #40000,R0 ;SET VIRTUAL ADDRESS TO THE START OF PAGE 2
2508 015066 012703 000400 MOV R2,KIPAR2 ;MAP PAGE 2 TO PHYSICAL ADDRESS
2509 015072 011420 58: MOV #256,,R3 ;SET UP COUNTER
2510 015074 005303 68: MOV (R4),(R0)+ ;LOAD DATA
2511 015076 001375 DEC R3
2512 015100 062702 000010 ADD #10,R2 ;AFTER EACH 256 WORDS, REMAP
2513 015104 020237 001206 CMP R2,HCOR12 ;DONE?
2514 015110 002403 BLT B7 ;NO= BRANCH
2515 015112 004737 012256 88: JSR PC,RST04
2516 015116 000207 RTS PC
2517 015120 005724 78: TST (R4)+ ;YES= EXIT
2518 015122 020427 015232 CMP R4,#WCASEE ;MOVE DATA POINTER
2519 015126 103750 BLO B8 ;TO NEXT WORST CASE VALUE
2520 015130 000745 BR B8 ;AND CONTINUE
2521
2522 015132 177776 000001 177775 WCASE: .WORD 177776,1,177775,2,177773,4,177767,10
2523 015140 000002 177773 000004
2524 015146 177767 000010
2525 015152 177757 000020 177737 .WORD 177757,20,177737,40,177677,100,177577,200
2526 015160 000040 177677 000100
2527 015166 177577 000200
2528 015172 177377 000400 176777 .WORD 177377,400,176777,1000,175777,2000,173777,4000
2529 015200 001000 175777 002000
2530 015206 173777 004000
2531 015212 167777 010000 157777 .WORD 167777,10000,157777,20000,137777,40000,77777,100000
2532 015220 020000 137777 040000
2533 015226 077777 100000
2534 015232 WCASEE:
    
```

```

2535 ;COMMAND TABLE. KEYBOARD COMMANDS ARE MATCHED AGAINST IT.
2536 ;THE MATCH MUST BE EXACT. NO ABBREVIATIONS ARE ALLOWED.
2537
2538 015232 COMTAB:
2539 015232 TOKN **"RUN",RUN
2540 015232 052522 000116 .ASCIZ %RUN%
2541 015240 007420 .WORD PUN
2542 015242 TOKN **"MOD",MOD
2543 015242 047515 000104 .ASCIZ %MOD%
2544 015250 010574 .WORD MOD
2545 015252 TOKN **"SEL",SEL
2546 015252 042523 000114 .ASCIZ %SEL%
2547 015260 010422 .WORD SEL
2548 015262 TOKN **"DES",DES
2549 015262 042504 000123 .ASCIZ %DES%
2550 015270 010440 .WORD DES
2551 015272 TOKN **"MAP",MAP
2552 015272 040515 000120 .ASCIZ %MAP%
2553 015300 010254 .WORD MAP
2554 015302 TOKN **"FILL",FILL
2555 015302 044506 046114 000 .ASCIZ %FILL%
2556 015310 010566 .WORD FILL
2557 015312 TOKN **"KTOFF",KTOFF
2558 015312 052113 043117 000106 .ASCIZ %KTOFF%
2559 015322 010522 .WORD KTOFF
2560 015324 TOKN **"KTON",KTON
2561 015324 052113 047117 000 .ASCIZ %KTON%
2562 015332 010522 .WORD KTON
2563 015334 TOKN **"PON",PON
2564 015334 047520 000116 .ASCIZ %PON%
2565 015342 010522 .WORD PON
2566 015344 TOKN **"POFF",POFF
2567 015344 047520 043106 000 .ASCIZ %POFF%
2568 015352 010522 .WORD POFF
2569 015354 TOKN **"ROTON",ROTON
2570 015354 047522 047524 000116 .ASCIZ %ROTON%
2571 015364 010526 .WORD ROTON
2572 015366 TOKN **"ROTOFF",ROTOFF
2573 015366 047522 047524 043106 .ASCIZ %ROTOFF%
2574 015374 000
2575 015376 010542 .WORD ROTOFF
2576 015400 015 000 .BYTE 15,0 ;CR
2577 015402 004 000 .BYTE 4,0
2578 015404 007364 .WORD COMCD2
2579 015406 000000 000000 000000 .WORD 0,0,0
    
```

```

2580                                     ;IMPURE MESSAGES.
2581
2582 015414 020045 020040 020040 AMODNM; .ASCII "% AT "
2583 015422 040440 020124 020040
2584 015426 020040 020040 020040 APC; .ASCII " STAT "
2585 015434 051440 040524 020124
2586 015442 020040 020040 020040 AMDSTA; .ASCII " "
2587 015450 020040 040520 041523 MDIRE; .ASCII " PASCNT "
2588 015456 052116 040 040
2589 015461 040 020040 020040 APSCNT; .ASCII " ERRCNT "
2590 015466 020056 042440 051122
2591 015474 047103 020124 020124
2592 015500 020040 020040 027040 AERRS; .ASCIZ " ."
2593 015506 000 000
2594
2595 015507 040 020040 020040 .ODD
        HOLD5; .ASCIZ " "
2596 015514 000 000
2597 015515 040 020040 020040 HOLD6; .ASCIZ " "
2598 015522 020040 000040 000040
2599 015526 020040 020040 020040 HOLD8; .ASCIZ " "
2600 015534 020040 020040 000
2601 015541 040 020040 020040 AEND; .ASCII " PC="
2602 015546 020040 041520 055
2603 015552 040 020040 020040 AEND1; .ASCII " APC="
2604 015560 020040 050101 026503
2605 015566 020040 020040 020040 AEND2; .ASCIZ " "
2606 015574 000040 000040
2607 015576 020040 040520 051523 $PASS; .ASCII " PASS"
2608 015604 020043 000040
2609 015606 020040 020040 027040 $PASS; .ASCIZ " ."
2610 015614 000 000
2611 015615 040 051104 050117 $DDEWD; .ASCIZ " DROPPED"
2612 015622 042520 022504 000
2613 015627 105 042116 040520 $SEND; .ASCII "ENDPAS="
2614 015632 026523 000040
2615 015636 020040 020040 027040 $PSCNT; .ASCIZ " ."
2616 015644 000 000
2617 015645 045 051503 040522 $ERROR; .ASCII "%CSPA "
2618 015652 040 040
2619 015653 040 020040 020040 $CSRAC; .ASCII " CSRC "
2620 015660 020040 051503 041522
2621 015666 040 040
2622 015667 040 020040 020040 $CSRC; .ASCII " STATC "
2623 015674 020040 052123 052101
2624 015702 020103 000040
2625 015704 020040 020040 020040 $STATC; .ASCIZ " "
2626 015712 000040 000040
2627 015714 041445 051123 020101 $DTERR; .ASCII "%CSRA "
2628 015722 020040 020040 020040 $DTE6; .ASCII " S/B "
2629 015730 051440 041057 040
2630 015735 040 020040 020040 $DTE3; .ASCII " WAS "
2631 015742 020040 040527 020123
2632 015750 020040 020040 020040 $DTE2; .ASCII " WRADR "
2633 015756 053440 040522 051104
2634 015764 040 040
2635 015765 040 020040 020040 $DTE5; .ASCII " RDADR "
    
```

```

2636 015772 020040 042122 042101
2637 016000 020122 000040
2638 016002 020040 020040 020040 ADTE4; .ASCIZ " "
2639 016010 000040 000040
2640 016012 042040 052101 020101 ADTE4A; .ASCIZ " DATA ERROR"
2641 016020 051105 047522 000122
2642 016026 047527 042122 020040 ADTEX; .ASCII "WORD"
2643 016034 020040 020040 027040 ADTE7; .ASCIZ "% "
2644 016042 000045 000045
2645 016044 042440 051122 020043 ERRNMB; .ASCII " ERR"
2646 016052 020040 020040 027040 AERNMB; .ASCIZ " ."
2647 016060 000 000
2648 016061 045 042515 020115 MERR1; .ASCII "%MEM FAILURE- PA "
2649 016066 045 046111 051125
2650 016074 026505 050040 020101
2651 016102 020040 020040 020040 MERR2; .ASCII " S/B "
2652 016110 051440 041057 040
2653 016115 040 020040 020040 MERR3; .ASCII " WAS "
2654 016122 020040 040527 020123
2655 016130 020040 020040 020040 MERR4; .ASCIZ " "
2656 016136 000040 000040
2657 016140 020045 020040 020040 AERNM; .ASCII "% HAD "
2658 016146 044040 042101 040
2659 016153 040 020040 020040 AFRCT; .ASCII " DATA EPRORS OUT OF "
2660 016160 020056 040504 040524
2661 016166 042440 051122 051117
2662 016174 020123 052517 020124
2663 016202 043117 040 040
2664 016205 040 020040 020040 AERTOT; .ASCIZ " WORDS READ"
2665 016212 020056 047527 042122
2666 016220 020123 042522 042101
2667 016226 000045 000045
2668 016230 040 040 040 DECVAL; .BYTE 40,40,40,40,40,40
2669 016233 040 040 040
    
```

```

2670 ;BUFFER AREAS,
2671
2672 .EVEN
2673 MODQ: ;ROOM FOR 20 MODULE POINTERS
2674 KBUF: ;KEYBOARD BUFFER, KBUFL LONG
2675 .BLKW 64. ;MONITOR STACK AREA
2676 SPBOT:
2677 .BLKW 16. ;SPECIAL MONITOR STACK AREA
2678 SPSPEC:
2679 IOQ: .BLKB IOQL
2680 TYPEQ: .BLKB TYPQL
2681
2682 ;*,
2683 BUFSIZ=AMODNM-START ;FIRST ADDRESS AFTER QUEUES
2684

```

```

2685 ;ROUTINE TO DETERMINE WHETHER WRITE BUFFER ROTATION SHOULD TAKE PLACE,
2686 ;AND TO DETERMINE CORE LIMITS OF BUFFER ROTATION. ALSO TO DETERMINE USE OF
2687 ;RTI OR RTT INSTRUCTION.
2688 ;*IOQ
2689 016606 012737 016624 000010 SETBUF: MOV #18,RESIV ;SET UP A RESERVED INSTRUCTION TRAP.
2690 016614 005046 CLR -(SP) ;CLEAR A WORD ON THE STACK FOR A NEXT PS*
2691 016616 012746 016654 MOV #28,-(SP) ;SET UP TO EXIT WITH RTT INSTRUCTION.
2692 016622 000006 RTT ;IF RTT NOT VALID IT WILL TRAP OUT.
2693 016624 012737 000002 005000 18: MOV #PTI,TRCIA ;TRAP COMES HERE. CHANGE RTT'S TO PTI'S.
2694 016632 012737 000002 002620 MOV #RTI,RTTI ; BY SHIFTING THE INSTRUCTION VALUE.
2695 016640 013737 001336 001346 MOV CHK2,CHK1 ;CHANGE SET OF CHKSUM WORDS BECAUSE
2696 016646 013737 001340 001350 MOV CHK2,CHK1 ;OF RTI'S REPLACING RTT'S
2697 ;OF ALL RTT'S TO THOSE OF RTI'S
2698 016654 012737 012524 000010 28: MOV #RESINT,RESIV ;RESTORE THE RESERVED INSTRUCTION TRAP VECTOR
2699 016662 105037 001301 CLR MDL45 ;NOT A 45
2700 016666 105037 001252 CLRB KTPRES ;NO KTI1
2701 016672 105037 001261 STBFA: CLR ROTI ;ASSUME NO BUFFER ROTATION.
2702 016676 105037 001263 CLR PFLC ;ASSUME NO CODE RELOCATION
2703 016702 013737 000000 001200 MOV 0,LOCORE ;GET LOCORE ADDR. FROM LOC 0
2704 016710 001004 BNE 16 ;BR IF LINKED WITH CONFIG/LINKER PROGRAM
2705 ;IF NOT ASSUMES NOT IN CHAIN MODE
2706 016712 005137 001200 COM LOCORE ;INDICATE SETBUF HAS BEFN RUN
2707 016716 000137 017214 JMP SFTXT ;GO PREPARE TO EXIT SETBUF ROUTINE
2708 016722 004537 012174 18: JSR R5,TOP12S ;GET TOP 12 BITS OF LOCOPE PHYS ADDR
2709 016726 001200 LOCORE ;AND PUT IT INTO LCOR12
2710 016730 032737 000777 001200 BIT #777,LOCORE ;IS IT ON A 256 WORD BOUNDARY ?
2711 016736 001414 REQ 28 ;YES, CONTINUE
2712 016740 042737 000777 001200 RIC #777,LOCORE ;NO, CLEAR OUT LOWER BITS
2713 016746 042737 000007 001204 BIC #7,LCOR12 ;ROUNDING OFF TO NEXT HIGHEST 256 BOUNDARY
2714 016754 062737 001000 001200 ADD #1000,LOCORE ;ROUND UP TO THE NEXT 256 WORD BOUNDARY
2715 016762 062737 000010 001204 ADD #10,LCOR12 ;ALSO INDICATE THAT BOUNDARY IN LOCORE12
2716 016770 013737 000042 001206 28: MOV #42,HCOR12 ;ARE WE IN CHAIN MODE?
2717 016776 001403 BEQ KTTST ;IF NO,GO DETERMINE IF KT IS TO BE USED
2718 017000 112737 177777 001260 MOVB #1,CHN ;IF YES, SET THE CHAIN INDICATOR
2719 017006 105737 001252 KTTST: TSTB KTPRES ;IS THE KTI1 AVAILABLE FOR USE?
2720 017012 001035 BNE SETKT ;IF YES, GO SIZE MEMORY USING IT
2721 017014 012737 017056 000004 MOV #58,4 ;IF NO, ONLY SIZE MEMORY UP TO 28K
2722 017022 012700 174000 MOV #-4000,R0 ;INITIALIZE LOOP COUNTER TO POINT TO BANK 0
2723 017026 005001 CLR R1 ;INITIALIZE THE COUNTER OF 1K CHUNKS(BANKS)
2724 017030 062700 004000 48: ADD #4000,R0 ;POINT TO THE FIRST WORD IN THE NEXT 1K BANK
2725 017034 062710 000000 ADD #0,(R0) ;WILL TRAP IF LOC DOESN'T EXIST
2726 017040 005201 INC R1 ;IF NO, CONTINUE PROCESSING AT 48
2727 ;IF YES,UP THE MEMORY BANK COUNT
2728 017042 022701 000034 CMP #28,,R1 ;HAVE 28 BANKS BEEN COUNTED?
2729 017046 001370 BNE 48 ;IF NO, GO CONTINUE SIZING
2730 017050 062700 004000 ADD #4000,R0 ;IF YES, INDICATE THE NUMBER OF WORDS IN R0
2731 017054 000401 BR 68 ;SKIP OVER THE NEXT INSTRUCTION
2732 017056 022626 58: CMP (SP)+,(SP)+ ;IF THERE WAS NO LOCATION AT LAST TRY,
2733 ;REMOVE THE TRAP FROM THE STACK
2734 017060 000300 68: SWAB R0 ;ISOLATE THE 12 HIGH ORDER BITS. FIRST,
2735 ;GET THE MOST SIGNIFICANT BITS IN THE LOW BYTE
2736 017062 006300 ASL R0 ;ALIGN THE BITS TO ACCOUNT FOR
2737 017064 006300 ASL R0 ;EXTENDED ADDRESS BITS (17 + 18)
2738 017066 042700 170037 BIC #170037,R0 ;ELIMINATE THE UNWANTED BITS FROM R0
2739 017072 105737 001260 TSTB CHN ;ACT11 CHAIN?
2740 017076 100003 BPL SETCOM ;NO- DON'T ADD ROOM FOR MONITOR

```

2741	017100	162700	000100			SUB	#100,R0	;YES- ALLOW ROOM FOR MONITOR
2742	017104	000400				BR	SETCOM	;GO TEST THE MEMORY
2743	017106					SETXT;		
2744	017106	012737	012516	000004		SETCOM;	MOV #BUSERR,4	;RESET BUS ERROR TRAP.
2745	017114	010037	001206			MOV	R0,HCOR12	;HI ADDR TO HCOR12.
2746	017120	004737	017222			SETCM1;	PC,TSTMEM	;TEST MEMORY FOR BASIC FUNCTIONING
2747								;VIA ADDRESS UP/DOWN TEST
2748	017124	013700	001206			MOV	HCOR12,R0	;DIFF BETWEEN LCOR12 AND HCOR12
2749	017130	010037	001210			MOV	R0,HCOR8V	;INITIALIZE SAVE LOCATION FOR KTON/OFF COMMANDS
2750	017134	163700	001204			SUB	LCOR12,R0	;MUST BE AT LEAST 40
2751	017140	022700	000040			CMF	#40,R0	;IS IT?
2752	017144	101023				BHI	SETXT	;FORGET IT IF NOT.
2753	017146	105237	001261			INCB	ROTI	;ALLOW ROTATION.
2754	017152	105737	001252			TSTB	KTPRES	;NO RELOCATION IF NO KT11
2755	017156	001413				BEG	18	
2756	017160	022700	000400			CMF	#400,R0	;IF 8K OR GREATER, ALLOW CODE RELOCATION
2757	017164	101010				BHI	18	;BRANCH IF NOT
2758	017166	013737	001204	001220		MOV	HCOR12,OFFMX	;CALCULATE THE HIGHEST LOC TO WHICH
2759	017174	163737	001204	001220		SUB	LCOR12,OFFMX	;THE PROGRAM CAN BE RELOCATED
2760	017202	105237	001263			INCB	RELOC	;ENABLE RELOCATION
2761	017206	004737	015016		18;	JSR	PC,LOADWC	;LOAD WORST CASE UNIBUS NOISE PATTERN
2762	017212	000005				RESET		;TURN OFF KT11 IF IN USE
2763	017214					SETXT;		
2764	017214	005037	000000			CLR	0	;CLEAR LOC 0,
2765	017220	000205				PTS	25	;EXIT.

2766								;MEMORY ADDRESS UP/DOWN TEST
2767								
2768	017222	023737	001206	001204		TSTMEM;	CMF HCOR12,LCOR12	;ANY MEMORY TO TEST ?
2769	017230	001001				BNE	58	;YES, CONTINUE
2770	017232	000207				RTS	PC	;NO, RETURN
2771	017234	013702	001204		58;	MOV	LCOR12,R2	;LOAD 12 BIT LOCOPE ADDRESS
2772	017240	013701	001200			MOV	LOCOPE,R1	;R1 CONTAINS LOCOPE PHYS ADDRESS
2773	017244	012737	017626	000004		MOV	#NOMEM,4	;LOAD TIME-OUT VECTOR
2774	017252	105737	001252		18;	TSTR	KTPRES	;USE KT11?
2775	017256	001002				BNE	28	;YES- BRANCH
2776	017260	010100				MOV	R1,R0	;NO- SET VA EQUAL TO PA
2777	017262	000404				BR	38	
2778	017264	012700	040000		28;	MOV	#40000,R0	;SET VA TO PAGE 2
2779	017270	010237	172344			MOV	R2,KIPAR2	;MAP PAGE 2 TO PHYSICAL ADDRESS
2780	017274	012703	000040		38;	MOV	#32,,R3	;COUNT 1 BLOCK
2781	017300	010120			48;	MOV	R1,(R0)+	;LOAD LOCATION WITH ITS ADDRESS
2782	017302	062701	000002			ADD	#2,R1	
2783	017306	005303				DEC	R3	
2784	017310	001373				BNE	46	
2785	017312	005202				INC	R2	;AFTER 1 BLOCK, UP PAGE ADDRESS FIELD
2786	017314	020237	001206			CMF	R2,HCOR12	;AT TOP?
2787	017320	002754				BLT	18	;NO- CONTINUE
2788	017322	005302				DEC	R2	;YES- TEST GOING DOWN
2789	017324	162701	000002			SUB	#2,R1	
2790	017330	105737	001252		TSTM1;	TSTB	KTPRES	;USE KT11?
2791	017334	001002				BNE	18	;YES- BRANCH
2792	017336	010100				MOV	R1,R0	;NO- LOAD VIRTUAL ADDRESS POINTER WITH PHYS ADDRESS
2793	017340	000404				BR	28	
2794	017342	012700	040076		18;	MOV	#40076,R0	;MAP VA TO TOP OF 1ST BLOCK IN PAGE 2
2795	017346	010237	172344			MOV	R2,KIPAR2	;MAP PAGE 2 TO PA
2796	017352	012703	000040		28;	MOV	#32,,R3	;COUNT 1 BLOCK
2797	017356	011037	001176		38;	MOV	#R0,MEMWAS	
2798	017362	020137	001176			CMF	R1,MEMWAS	;DOES LOCATION CONTAIN ITS PA?
2799	017366	001120				BNE	MERR	;NO- BRANCH
2800	017370	162700	000002		48;	SUB	#2,R0	
2801	017374	162701	000002			SUB	#2,R1	
2802	017400	005303				DEC	R3	
2803	017402	001365				BNE	38	
2804	017404	005302				DEC	R2	;AFTER CHECKING 1 BLOCK, ADDRESS NEXT
2805	017406	020237	001204			CMF	R2,LCOR12	;DONE?
2806	017412	002346				BGE	T8YM1	;NO- CONTINUE CHECKING
2807	017414	013702	001206			MOV	HCOR12,R2	;YES- SET TO LOAD MEMORY
2808	017420	010201				MOV	R2,R1	;FROM TOP DOWN
2809	017422	005302				DEC	R2	
2810	017424	006301				ASL	R1	
2811	017426	006301				ASL	R1	
2812	017430	006301				ASL	R1	
2813	017432	006301				ASL	R1	
2814	017434	006301				ASL	R1	
2815	017436	005301				DEC	R1	
2816	017440	006301				ASL	R1	
2817	017442	105737	001252		TSTM2;	TSTB	KTPRES	;USE KT11?
2818	017446	001002				BNE	18	;YES- BRANCH
2819	017450	010100				MOV	R1,R0	;NO- LOAD VIRTUAL POINTER WITH
2820								;PHYSICAL ADDRESS
2821	017452	000404				BR	28	

```

2822 017454 012700 040076      18:  MOV      #40076,R0      ;LOAD VIRTUAL WITH TOP ADDRESS OF
2823                                     ;1ST BLOCK IN PAGE 2
2824 017460 010237 172344      28:  MOV      R2,KIPAR2     ;MAP PAGE 2 TO PHYSICAL
2825 017464 012703 000040      38:  MOV      #32,,R3      ;COUNT A BLOCK
2826 017470 010110                                     ;LOAD LOCATION WITH ITS PHYS ADDRESS
2827 017472 005110                                     ;COMPLEMENT IT
2828 017474 162701 000002      COM      #2,R1
2829 017500 162700 000002      SUB      #2,R0
2830 017504 005303      DEC      R3
2831 017506 001370      BNE     38
2832 017510 005302      DEC      R2      ;AFTER LOADING A BLOCK, SET UP FOR NEXT
2833 017512 020237 001204      CMP      R2,LCOR12  ;DONE?
2834 017516 002351      BGE     TSTM2     ;NO- BRANCH
2835 017520 005202      INC      R2      ;YES- SETUP TO CHECK IT
2836 017522 062701 000002      ADD      #2,R1
2837 017526 010104      MOV      R1,R4
2838 017530 105737 001252      TSTM3:  TSTB     KTPRES  ;USE KT11?
2839 017534 001002      BNE     18      ;YES- BRANCH
2840 017536 010400      MOV      R4,R0      ;SET VIRTUAL ADDRESS EQUAL TO PHYS ADDRESS
2841 017540 000404      BR      28
2842 017542 012700 040000      18:  MOV      #40000,R0   ;SET VIRTUAL ADDRESS TO START OF PAGE 2
2843 017546 010237 172344      MOV      R2,KIPAR2   ;MAP PAGE 2 TO PHYSICAL ADDRESS
2844 017552 012703 000040      28:  MOV      #32,,R3    ;COUNT A BLOCK
2845 017556 011037 001176      38:  MOV      #R0,MEMWAS
2846 017562 010401      MOV      R4,R1
2847 017564 005101      COM      R1
2848 017566 020137 001176      CMP      R1,MEMWAS   ;DOES LOCATION CONTAIN COMPLEMENT OF ITS PHYS ADDRESS?
2849 017572 001016      BNE     MERR        ;NO- BRANCH
2850 017574 005720      48:  TST      (R0)+
2851 017576 062704 000002      ADD      #2,R4
2852 017602 005303      DEC      R3
2853 017604 001364      BNE     38
2854 017606 005202      INC      R2      ;CHANGE 12 BIT ADDRESS AFTER CHECKING A BLOCK
2855 017610 020237 001206      CMP      R2,HCOR12  ;DONE?
2856 017614 002745      BLT     TSTM3     ;NO- BRANCH
2857 017616 012737 012516 000004      MOV      #BUSERR,4  ;RESET THE TIME-OUT TRAP VECTOR
2858 017624 000207      RTS                                     ;YES- RETURN
2859
2860 017626 022626      NOMEM:  CMP      (SP)+,(SP)+      ;POP THE STACK POINTER TWICE
2861 017630 012737 012516 000004      MERR:  MOV      #BUSERR,4      ;RESET THE TIME-OUT TRAP VECTOR
2862 017636 105237 001303      INCB     MEMERF
2863 017642 010037 001164      MOV      R0,MEMVA   ;SAVE FAILING ADDRESS
2864 017646 104413 001164      GETPA8, MEMVA
2865 017652 010137 001174      MOV      R1,MEMSBE  ;SAVE EXPFCTED CONTENTS
2866 017656 013746 001166      MOV      MEMPA,-(SP) ;CONVEPT PHYSICAL ADDRESS TO ASCII
2867 017662 013746 001170      MOV      MEMEA,-(SP)
2868 017666 004537 011524      JSR     R5,OACNVX
2869 017672 016102      MERR2
2870 017674 013746 001174      MOV      MEMSBE,-(SP) ;CONVERT SHOULD BE TO ASCII
2871 017700 004537 011512      JSR     R5,OACNV
2872 017704 016115      MERR3
2873 017706 013746 001176      MOV      MEMWAS,-(SP) ;CONVERT WAS VALUE TO ASCII
2874 017712 004537 011512      JSR     R5,OACNV
2875 017716 016130      MERR4
2876 017720 004537 012174      JSR     R5,TOP128   ;GET NEW HIGH CORE VALUE
2877 017724 001166      MEMPA
    
```

```

2878 017726 013737 001172 001206      MOV      MEM12,HCOR12
2879 017734 042737 000007 001206      BIC     #7,HCOR12
2880 017742 162737 000040 001206      SUB     #40,HCOR12
2881 017750 000207      RTS     PC
    
```











