```
.REM _
```

IDENTIFICATION

PRODUCT CODE: AC-E860F-MC

PRODUCT NAME: CXPAAF0 PA611 READER MODULE

PRODUCT DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

1. ARSTRACT:

PAA IS AN IOMOD THAT EXERCISES UP TO 16 HIGH SPEED PAPER TAPE READERS BY READING A STANDARD BINARY COUNT PATTERN. IT REPORTS ALL ERRORS ON THE CONSOLE TTY. THE MODULE IS DESIGNED TO ACTIVATE ALL DEVICES AND READ WITH ALL ACTIVE DEVICES RUNNING CONCURRENTLY.

2. REQUIREMENTS:

HARDWARE: AT LEAST ONE PA611-R CONTROL UNIT AND ONE

PR69E HIGH SPEED READER

STORAGE:: PAA REQUIRES:
1. DECIMAL WORDS: 404
2. OCTAL WORDS: 0624
3. OCTAL BYTES: 1450

3. PASS DEFINITION:

ONE PASS OF THE PAA MODULE CONSISTS OF READING 2048. CHARACTERS

4. EXECUTION TIME:

PAA RUNNING ALONE ON A PDP11/05 PROCESSOR TAKES APPROXIMATELY---MINUTES TO COMPLETE ONE PASS.

5. CONFIGURATION REQUIREMENTS:

DEFAULT PARAMETERS:

DEVADR: 172600, VECTOR: 300, BR1: 4, DEVCNT: 1

REQUIRED PAPAMETERS:

NONE

6. DEVICE/OPTION SET-UP:

A. TURN ON ALL PR68E READERS TO BE TESTED
B. LOAD EACH READER WITH A BINARY COUNT TAPE AND POSITION THE TAPE SUCH THAT A NON-ZERO PUNCH IS OVER THE READER PHOTO CELLS (A FABRICATED TEST LOOP TAPE IS RECOMMENDED TO ELIMINATE THE NEED TO RELOAD)

7. MODULE OPERATION:

TEST SEQUENCE:

A. TEST UP TO 16 POSSIBLE DEVICES FOR SELECTION
B. STORE THE DEVICE NO. OF THOSE DEVICES SELECTED TO BE TESTED AND SET UP THEIR APPROPRIATE VECTOR
C. TURN ON THE INTERRUPT ENABLE AND READER ENABLE FOR ALL ACTIVE DEVICES.

D. INTERRUPT SERVICE

1. READ 64 CHARACTERS TO SYNCHRONIZE THE DATA PATTERN BEFORE TESTING FOR ERRORS
2. TEST FOR AND REPORT ANY ERROR BITS IN THE CSR
3. COMPARE THE DATA READ WITH THE EXPECTED DATA
4. REPORT ANY DATA COMPARISION ERRORS
5. IF NO DATA ERRORS GO TO STEP 7
6. IF DATA ERROR RESYNC THE DATA
7. ENABLE THE READER AND INTERRUPT TO READ THE NEXT FRAME
8. REPEAT STEPS 2-7 UNTIL 2096. FRAMES HAVE BEEN PROCESSED

E. TURN OFF ALL ACTIVE DEVICES AND REPORT END OF PASS
F. RESTART AT STEP C

JSR TABLE:

TO LINK THE INDIVIDUAL INTERRUPTS WITH THE SERVICE ROUTINES THERE IS A JSR TABLE CONTAINING 16 ENTRIES. EACH DEVICE VECTOR IS SET UP TO POINT TO A UNIQUE JSR WITHIN THE TABLE WHICH TRANSFERS CONTROL TO THE SERVICE ROUTINE AND POINTS AS TO AN OFFSET THAT THE SERVICE ROUTINE WILL USE TO GENERATE THE CORRESPONDING REGISTER ADDRESS AND DATA TABLE ENTRY FOR THAT LINE.

FIFO QUEUE:

TO ALLOW THE SERVICE ROUTINES TO USE THE SAME GPRS FOR SERVICING UP TO 16 CONCURRENT INTERRUPTS FROM DIFFERRENT LINES, THE INTR. REQUEST IS STORED IN A FIFO QUEUE TO BE SERVICED LATER. THE QUEUE HAS 16 ENTRIES AND THE INTERRUPT SERVICE ROUTINE STORES THE OFFSET NO. OF THE LINE IN THE QUEUE AND THEN EXECUTES A "PIRQ" CALL.
WHEN THE "PIRQ" CALL IS SERVICED THE OFFSET IS RETRIEVED FROM THE QUEUE AND THE DEVICE SERVICED. TWO POINTERS ARE REQUIRED TO SERVICE THE QUEUE:

QPTR1	USED TO STORE THE OFFSET
QPTR2	USED TO RETRIEVE THE OFFSET

ERROR RETURN QUEUE:

WHEN AN "ERROR" OR A "DATERR" CALL IS EXECUTED THE MONITOR QUEUES THE RETURN ADDRESS IN ITS "IOQUE". SINCE IT IS POSSIBLE THAT ANOTHER LINE MAY ENJOY A HIGHER PRIORITY IN THE MONITOR'S QUEUE, IT COULD GET CONTROL OF THE MODULE'S PIRQ SERVICE ROUTINE AND CLOBBER THE REGISTERS. WHEN THE ERROR LINE RETURNS IT ENDS UP WITH THE WRONG REGISTER ADDRESSES. TO PREVENT THIS FROM HAPPENING, A SECOND FIFO QUEUE IS USED TO QUEUE RETURNS FROM ERRORS. PRIOR TO ANY ERROR CALL THE INFORMATION NECESSARY TO RESTORE THE CORRECT ADDRESS IS STORED IN THE QUEUE AND WHEN CONTROL IS RETURNED AFTER THE ERROR THIS INFORMATION IS RETRIEVED. THE QUEUE CONSISTS OF 16 BYTE ENTRIES AND REQUIRES THE FOLLOWING POINTERS FOR ITS OPERATION:

REQP1	SAVES R1 IN THE QUEUE
REQP2	RETRIEVES R1 FROM THE QUEUE

8. OPERATION OPTIONS:

A. LOCATION DVID1 MAY BE ALTERED TO CHANGE THE NUMBER AND COMBINATION OF DEVICES TO BE TESTED.
BIT0=DEV0,BIT1=DEV1.....BIT15=DEV15.

9. NON-STANDARD PRINTOUTS:

NONE: ALL PRINTOUTS HAVE THE STANDARD MEANING AND FORMAT DESCRIBED IN THE DEC/X11 DOCUMENT

```
                ;PA611R DEC/X11 EXERCISER MODULE

000000'                 IOMOD   <PAAF >,172600,300,4,,2048.,53
000000'                 MODULE  140000,PAAF ,172600,300,4,,,2048.,53
                        .TITLE  PAAF DEC/X11 SYSTEM EXERCISER MODULE
                ;       DDXCOM  VERSION  6      23-MAY-78
                                .CLIST  BIN
                        ;*****************************************************************************
000000'                 ASGIN:
000000' 040520  043101      040 MODNAM: .ASCII  /PAAF / ;MODULE NAME.
000004'     000                 XFLAG:  .BYTE   0PEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 172600                  ADDR:   172600+0                ;1ST DEVICE ADDR.
000010' 000300                  VECTOR: 300+0                   ;1ST DEVICE VECTOR.
000012'     200                 BR1:    .BYTE   PRTY4+0         ;1ST BR LEVEL.
000013'     000                 BR2:    .BYTE   PRTY+0          ;2ND BR LEVEL.
000014' 000001                  DVID1:  +1                      ;DEVICE INDICATOR 1.
000016' 000000                  SR1:    OPEN                    ;SWITCH REGISTER 1
000020' 000000                  SR2:    OPEN                    ;SWITCH REGISTER 2
000022' 000000                  SR3:    OPEN                    ;SWITCH REGISTER 3
000024' 000000                  SR4:    OPEN                    ;SWITCH REGISTER 4
                        ;*****************************************************************************
000026' 140000                  STAT:   140000                  ;STATUS WORD.
000030' 000224'                 INIT:   START                   ;MODULE START ADDR.
000032' 000224'                 SPOINT: MODSP                   ;MODULE STACK POINTER.
000034' 000000                  PASCNT: 0                       ;PASS COUNTER.
000036' 004000                  ICONT:  2048.                   ;# OF ITERATIONS PER PASS=2048.
000040' 000000                  ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
000042' 000000                  SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
000044' 000000                  HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
000046' 000000                  SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
000050' 000000                  HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
000052' 000000                  SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
000054' 000066                  RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056'                         CONFIG:                         ;RESERVED FOR MONITOR USE
000056' 000000                  RES1:   0                       ;RESERVED FOR MONITOR USE
000060' 000000                  RES2:   0                       ;RESERVED FOR MONITOR USE
000062' 000000                  SVR0:   OPEN                    ;LOC TO SAVE R0.
000064' 000000                  SVR1:   OPEN                    ;LOC TO SAVE R1.
000066' 000000                  SVR2:   OPEN                    ;LOC TO SAVE R2.
000070' 000000                  SVR3:   OPEN                    ;LOC TO SAVE R3.
000072' 000000                  SVR4:   OPEN                    ;LOC TO SAVE R4.
000074' 000000                  SVR5:   OPEN                    ;LOC TO SAVE R5.
000076' 000000                  SVR6:   OPEN                    ;LOC TO SAVE R6.
000100' 000000                  CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
000102'                         SRADR:                          ;ADDR OF GOOD DATA, OR
000102' 000000                  ACSR:   OPEN                    ;CONTENTS OF CSR.
000104'                         WASADR:                         ;ADDR OF BAD DATA, OR
000104' 000000                  ASTAT:  OPEN                    ;STATUS REG CONTENTS.
000106'                         ERRTYP:                         ;TYPE OF ERROR
000106' 000000                  ASB:    OPEN                    ;EXPECTED DATA.
000110' 000000                  AWAS:   OPEN                    ;ACTUAL DATA.
000112' 000240'                 RSTPT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114' 000500                  WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
```



```
000116' 000000                  WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
000120' 000000                  INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
000122' 000053                  IDNUM:  53                      ;MODULE IDENTIFICATION NUMBER=53
        000040                          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224'                         MODSP:
                        ;*****************************************************************************
                        ;MODULE INITIALIZATION
000224' 012767  004000  177666  START:  MOV     #2048.,INTR     ;2048 INTRS
000232' 012767  002000  177654          MOV     #1024.,WDTO     ;1024 WORDS TO
000240' 016767  177572  001156  RESTRT: MOV     ICONT,COUNT     ;DO APPROX. 2048 INTRS/PASS
000246' 012767  004000  001152          MOV     #2048.,COUNTC   ;KEEP TRACK OFF READS ISSUED
000254' 005067  001150                  CLR     GETOUT          ;RESET END OF PASS FLAG
000260' 012767  001272' 001146          MOV     #RQ,QPTR1       ;SET UP SERVICE QUEUE POINTERS
000266' 012767  001272' 001142          MOV     #RQ,QPTR2
000274' 012767  001312' 001136          MOV     #REQ,REQP1      ;SET UP ERROR RETURN QUEUE POINTERS
000302' 012767  001312' 001132          MOV     #REQ,REQP2
000310' 016700  177500                  MOV     DVID1,R0        ;GET DEVICE SELECTION PARAMETER
000314' 001002                          BNE     1$              ;BR IF ANY DEVICES SELECTED FOR TEST
000316' 104410  000000'                 ENDS,BEGIN              ;
000322' 005001          1$:     CLR     R1                      ;R1 AND R2 TO START AT ZERO
000324' 005101                  COM     R1
000326' 005002                  CLR     R2
000330' 005102                  COM     R2
000332' 016703  177452                  MOV     VECTOR,R3       ;START AT BEGINNING OF VECTOR AREA
000336' 012704  000452'                 MOV     #JSRTAB,R4      ;START AT BEGINNING OF JSR TABLE
000342' 005201          2$:     INC     R1                      ;COUNT A DEVICE
000344' 006200                  ASR     R0                      ;SHIFT SELECT BIT INTO "C"
000346' 103410                  BCS     4$                      ;BR IF SELECT BIT IS ON A "1"
000350' 062703  000004          ADD     #4,R3                   ;UPDATE VECTOR POINTER
000354' 062704  000006          ADD     #6,R4                   ;UPDATE JSR TABLE POINTER
000360' 020127  000017          CMP     R1,#17                  ;LAST POSSIBLE DEVICE
000364' 001512                  BEQ     STMP                    ;GO START UP ACTIVE DEVICES IF YES
000366' 000765                  BR      2$                      ;GO TEST FOR NEXT ONE
000370' 010423          4$:     MOV     R4,(R3)+                ;SET UP PRINTR POINTER
000372' 016423  177414          MOV     BR1,(R3)+               ;SET UP PRIORITY LEVEL
000376' 062704  000006          ADD     #6,R4                   ;UPDATE JSR TABLE POINTER
000402' 005202                  INC     R2                      ;COUNT ACTIVE DEVICE
000404' 110162  000412'         MOVB    R1,DEVTAB(R2)           ;STORE ACTIVE DEV.NO. IN TABLE
000410' 000763                  BR      3$                      ;GO TEST FOR LAST DEVICE

                        ;DEVICE AND DATA TABLES

000412' 000010          DEVTAB: .BLKW   8.              ;16 BYTE ACTIVE DEVICE TABLE
000432' 000010          DATTAB: .BLKW   8.              ;16 BYTE DATA TABLE


000452' 004567  000204  JSRTAB: JSR      R5,PRINTR      ;LINK TO READER 0
```

```
000456' 000000                         0
000460' 004567  000176                 JSR     R5,RINTR        ;LINK TO READER 1
000464' 000004                         4
000466' 004567  000170                 JSR     R5,RINTR        ;LINK TO READER 2
000472' 000010                         10
000474' 004567  000162                 JSR     R5,RINTR        ;LINK TO READER 3
000500' 000014                         14
000502' 004567  000154                 JSR     R5,RINTR        ;LINK TO READER 4
000506' 000020                         20
000510' 004567  000146                 JSR     R5,RINTR        ;LINK TO READER 5
000514' 000024                         24
000516' 004567  000140                 JSR     R5,RINTR        ;LINK TO READER 6
000522' 000030                         30
000524' 004567  000132                 JSR     R5,RINTR        ;LINK TO READER 7
000530' 000034                         34
000532' 004567  000124                 JSR     R5,RINTR        ;LINK TO READER 10
000536' 000040                         40
000540' 004567  000116                 JSR     R5,RINTR        ;LINK TO READER 11
000544' 000044                         44
000546' 004567  000110                 JSR     R5,RINTR        ;LINK TO READER 12
000552' 000050                         50
000554' 004567  000102                 JSR     R5,RINTR        ;LINK TO READER 13
000560' 000054                         54
000562' 004567  000074                 JSR     R5,RINTR        ;LINK TO READER 14
000566' 000060                         60
000570' 004567  000066                 JSR     R5,RINTR        ;LINK TO READER 15
000574' 000064                         64
000576' 004567  000060                 JSR     R5,RINTR        ;LINK TO READER 16
000602' 000070                         70
000604' 004567  000052                 JSR     R5,RINTR        ;LINK TO READER 17
000610' 000074                         74



                                ;INITIAL START UP ROUTINES

000612' 010267  000630          STUP:  MOV     R2,ACTDEV       ;SAVE ACTIVE DEVICE COUNT
000616' 012767  000100  000606         MOV     #64.,SYNFLG     ;FLAG TO INDICATE 64 CHARS. READ
000624' 116200  000412'         1$:    MOVB    DEVTAB(R2),R0   ;GET AN ACTIVE DEVICE NO.
000630' 010001                         MOV     R0,R1           ;DUPLICATE DEVICE NO. IN R1
000632' 006300                         ASL     R0              ;GENERATE CSR ADDRESS IN R0
000634' 006300                         ASL     R0
000636' 066700  177144                 ADD     ADDR,R0
000642' 052710  000101                 BIS     #101,(R0)       ;GO READ A CHAR, AND INTERRUPT
000646' 005367  000554                 DEC     COUNTC          ;COUNT A READ ISSUED
000652' 005302                         DEC     R2              ;COUNT IT
000654' 100363                         BPL     1$              ;KEEP GOING UNTIL ALL ACTIVE DEVICES
                                                               ;ARE KICKED OFF
000656' 104400  000000'                EXITS,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

                                ;READER INTERRUPT SERVICE ROUTINES

                                ;ALL INTERRUPTS ENTER HERE VIA JSR TABLE LINKAGE

000662' 111577  000546          RINTR: MOVB    (R5),@QPTR1     ;STORE REQUEST IN FIFO QUEUE
000666' 005267  000542                 INC     QPTR1           ;UPDATE Q POINTER
```



```
000672' 022767  001312' 000534         CMP     #RQ+20,QPTR1    ;QUEUE POINTER BEYOND LIMIT
000700' 001003                         BNE     1$              ;BR IF NOT
000702' 012767  001272' 000524         MOV     #RQ,QPTR1       ;RESET QUEUE POINTER
000710' 012605                  1$:    MOV     (SP)+,R5        ;CLEAN UP STACK
                                       ;-----------------------------------------------------------
000712' 000004  000000' 000720'        PIRQS,BEGIN,RDSERV              ; QUEUE UP TO CONTINUE AT RDSERV AND RTI
                                       ;-----------------------------------------------------------


                                ;ROUTINE TO SERVICE FIFO QUEUE

000720' 005767  000504          RDSERV: TST    GETOUT          ;END OF PASS?
000724' 100476                         BMI     ENPS            ;BR IF YES
000726' 117700  000504                 MOVB    @QPTR2,R0       ;GET OFFSET FROM Q
000732' 010001                         MOV     R0,R1           ;DUPLICATE OFFSET IN R1
000734' 005267  000476                 INC     QPTR2           ;UPDATE Q POINTER
000740' 022767  001312' 000470         CMP     #RQ+20,QPTR2    ;QUEUE POINTER BEYOND LIMIT
000746' 001003                         BNE     1$              ;BR IF NOT
000750' 012767  001272' 000460         MOV     #RQ,QPTR2       ;RESET POINTER
000756' 006201                  1$:    ASR     R1              ;GENERATE DEVICE NO. IN R1
000760' 006201                         ASR     R1
000762' 066700  177020                 ADD     ADDR,R0         ;GEN. CSR ADDRESS IN R0
000766' 005767  000440                 TST     SYNFLG          ;DATA TABLE SYNCHRONIZED?
000772' 001410                         BEQ     2$              ;BR IF YES (64 INTERRUPTS)
000774' 116061  000002  000432'        MOVB    2(R0),DATTAB(R1);LOAD RCVR,DBR INTO DATA TABLE
001002' 005367  000424                 DEC     SYNFLG          ;COUNT IT
001006' 005210                         INC     (R0)            ;READ ANOTHER FRAME
001010' 104400  000000'                EXITS,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
001014' 005710                  2$:    TST     (R0)            ;ERROR BIT IN CSR?
001016' 100004                         BPL     RSER0           ;BR IF NONE
001020' 012767  001030' 000416         MOV     #RSER0,FORK     ;SET UP RETURN TO RSER0
001026' 000452                         BR      RDERR           ;GO REPORT THE ERROR
001030' 105710                  RSER0: TSTB    (R0)            ;DONE SET ?
001032' 100404                         BMI     RSER1           ;BR IF SET
001034' 012767  001044' 000402         MOV     #RSER1,FORK     ;SET UP RETURN TO RSER1
001042' 000444                         BR      RDERR           ;GO REPORT THE ERROR
001044' 105261  000432'         RSER1: INCB    DATTAB(R1)      ;BUMP THE DATA CHAR. +1
001050' 126061  000002  000432'        CMPB    2(R0),DATTAB(R1);DID DATA IN DRB=DATA IN TABLE?
001056' 001401                         BEQ     RSER2           ;BR IF YES
001060' 000455                         BR      DATER           ;GO REPORT THE ERROR
001062' 005367  000336          RSER2: DEC     COUNT           ;COUNT THE INTERRUPT
001066' 001003                         BNE     1$              ;BR IF NOT 2048 YET
001070' 005167  000334                 COM     GETOUT          ;SET END OF PASS FLAG
001074' 000412                         BR      ENPS            ;GO TURN OFF ALL ACTIVE READERS
001076'                         1$:
001076' 104413  000000'                ENDITS,BEGIN            ;SIGNAL END OF ITERATION.
                                                               ;MONITOR SHALL TEST END OF PASS
001102' 005767  000320                 TST     COUNTC          ;HAVE WE ISSUED ENOUGH READS?
001106' 001403                         BEQ     EX              ;BR IF YES

001110' 005210                         INC     (R0)            ;READ ANOTHER FRAME
001112' 005367  000310                 DEC     COUNTC          ;COUNT ANOTHER READ ISSUED
001116'                         EX:
001116' 104400  000000'                EXITS,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

                                ;ROUTINE TO TURN OFF ALL ACTIVE READERS AFTER 2048. TRANSFERS
```

```
001122' 016701  000320'          ENPS:   MOV     ACTDEV,R1       ;GET NO OF ACTIVE DEVICES
001126' 116160  000412'          1$:     MOVB    DEVTAB(R1),R0   ;GET ACTIVE DEVICE NO. FM TABLE
001132' 006300                           ASL     R0              ;GEN. CSR ADDRESS
001134' 006300                           ASL     R0
001136' 066700  176644                   ADD     ADDR,R0
001142' 005010                           CLR     (R0)            ;TURN OFF READER
001144' 005301                           DEC     R1              ;COUNT IT
001146' 100367                           BPL     1$              ;BR UNTIL ALL OFF
001150' 104413  000000'                  ENDITS,BEGIN            ;SIGNAL END OF ITERATION.
                                                                 ;MONITOR SHALL TEST END OF PASS
                                                                 ;TO SHUT OFF INTERRUPTS

                                 ;ERROR REPORTING ROUTINES

001154' 010067  176720           RDERR:  MOV     R0,CSRA         ;SAVE CSR ADDRESS
001160' 011067  176716                   MOV     (R0),ACSR       ;SAVE CONTENTS OF CSR
001164' 004767  000142                   JSR     PC,QERR1        ;GO QUEUE RETURN FROM ERROR SERVICE
001170' 012767  000011  176710           MOV     #11,ERRTYP      ;ILLEGAL INTERRUPT OR DONE NOT SET
                                         ;*****************************************************************
001176' 104405  000000' 000000           RDERS,BEGIN,NULL        ;ERROR BIT SET OR FALSE INTERRUPT
                                         ;*****************************************************************
001204' 004767  000152                   JSR     PC,QERR2        ;GO GET DEVICE NO. FROM ERROR RETURN QUE
001210' 000177  000230                   JMP     @FORK           ;RETURN TO CALLER

001214' 010067  176660           DATER:  MOV     R0,CSRA         ;SAVE CSR ADDRESS
001220' 005720                           TST     (R0)+           ;GENERATE DBR ADDRESS IN R0
001222' 111067  176662                   MOVB    (R0),AWAS
001226' 116167  000432' 176652           MOVB    DATTAB(R1),ASR
001234' 010067  176644                   MOV     R0,WASADR
001240' 004767  000066                   JSR     PC,QERR1        ;GO QUEUE RETURN FROM ERROR SERVICE
001244' 062701  000432'                  ADD     #DATTAB,R1
001250' 010167  176626                   MOV     R1,SAADR
001254' 111011                           MOVB    (R0),(R1)
001256' 005740                           TST     -(R0)
                                         ;*****************************************************************
001260' 104404  000000'                  DATERS,BEGIN            ;DATA ERROR!!!
                                         ;*****************************************************************
001264' 004767  000072                   JSR     PC,QERR2        ;GO GET DEVICE NO. FROM ERROR RETURN QUE
001270' 000674                           BR      RSER2           ;RETURN TO READ ANOTHER FRAME

001272' 000010                   RQ:     .BLKW   8.              ;16 BYTE FIFO QUEUE
001312' 000010                   REQ:    .BLKW   8.              ;16 BYTE ERROR RETURN FIFO QUEUE

                                 ;SUBROUTINES TO QUEUE RETURN FROM ERROR SERVICE

001332' 110177  000102           QERR1:  MOVB    R1,@REQP1       ;SAVE THE DEVICE NO IN THE QUEUE
001336' 005267  000076                   INC     REQP1           ;UPDATE THE QUEUE POINTER
001342' 022767  001332' 000070           CMP     #REQ+20,REQP1   ;QUE POINTER AT HIGH LIMIT ??
001350' 001003                           BNE     1$              ;BR IF NOT
001352' 012767  001312' 000060           MOV     #REQ,REQP1      ;RESET THE POINTER
001360' 000207                   1$:     RTS     PC              ;RETURN TO REPORT THE ERROR

001362' 117701  000054           QERR2:  MOVB    @REQP2,R1       ;RETRIEVE THE DEVICE NO. FROM THE QUEUE
001366' 005267  000050                   INC     REQP2           ;UPDATE THE QUE POINTER
001372' 022767  001332' 000042           CMP     #REQ+20,REQP2   ;POINTER AT THE HIGH LIMIT ??
001400' 001003                           BNE     1$              ;BR IF NOT
001402' 012767  001312' 000032           MOV     #REQ,REQP2      ;RESET THE POINTER
```



```
001410' 010100                   1$:     MOV     R1,R0           ;DUPLICATE THE NO. IN R0
001412' 006300                           ASL     R0              ;GENERATE THE DEVICE REG OFFSET
001414' 006300                           ASL     R0
001416' 066700  176364                   ADD     ADDR,R0         ;REBUILD THE CSR ADDRESS IN R0
001422' 000207                           RTS     PC

                                 ;SOME MODULE VARIABLES

001424' 000000                   COUNT:  OPEN                    ;COUNTER FOR 2048 INTRS.
001426' 000000                   COUNTC: OPEN                    ;COUNTER FOR READS ISSUED
001430' 000000                   GETOUT: OPEN                    ;END PASS FLAG
001432' 000000                   SYNFLG: OPEN                    ;DATA TABLE SYNC FLAG
001434' 000000                   QPTR1:  OPEN                    ;FIFO QUEUE POINTERS
001436' 000000                   QPTR2:  OPEN
001440' 000000                   REQP1:  OPEN                    ;ERROR RETURN QUEUE PTRS
001442' 000000                   REQP2:  OPEN
001444' 000000                   FORK:   OPEN                    ;RETURN POINTER FOR ERROR RETURNS
001446' 000000                   ACTDEV: OPEN                    ;STORES NO. OF DEVICES ACTIVE

                 000001                  .END
```

```
ACSP    000102R          234#    424*
ACTDEV  001446R          334*    410     483#
ADDR    000006R          200#    340     374     414     469
ADDR22= 001000           252#
ASB     000106R          238#    436*
ASTAT   000104R          236#
AWAS    000110R          239#    435*
BEGIN   000000R          197#    267     346     359     380     398     406     418     428     444
BIT0    = 000001         252#
BIT1    = 000002         252#
BIT10   = 002000         252#
BIT11   = 004000         252#
BIT12   = 010000         252#
BIT13   = 020000         252#
BIT14   = 040000         252#
BIT15   = 100000         252#
BIT2    = 000004         252#
BIT3    = 000010         252#
BIT4    = 000020         252#
BIT5    = 000040         252#
BIT6    = 000100         252#
BIT7    = 000200         252#
BIT8    = 000400         252#
BIT9    = 001000         252#
BREAKS= 104407           252#
BR1     000012R          202#    283
BR2     000013R          203#
BTODS = 104421           252#
CDATAS= 104412           252#
CONFIG  000056R          222#
COUNT   001424R          258*    393*    474#
COUNTC  001426R          259*    342*    400     404*    475#
CSRA    000100R          232#    423*    433*
DATCKS= 104411           252#
DATER   001214R          397     433#
DATERS= 104404           252#    444
DATTAB  000432R          292#    377*    389*    390     436     439
DEVTAB  000412R          286*    291#    336     411
DVID1   000014R          204#    265
ENDITS= 104413           252#    398     418
ENDS    = 104410         252#    267
ENPS    001122R          395     396     410#
ERRTYP  000106R          237#    426*
EX      001116R          401     405#
EXITS = 104400           252#    346     380     406
FORK    001444R          383*    387*    431     482#
GETOUT  001430R          260*    364     395*    476#
GETPAS= 104415           252#
GWBUFS= 104414           252#
HRDCNT  000044R          217#
HRDERS= 104405           252#    428
HRDPAS  000050R          219#
ICONT   000036R          214#    258
ICOUNT  000040R          215#
IDNUM   000122R          244#
INIT    000030R          211#
```



```
INTR    000120R          243#    255*
JSRTAB  000452R          273     297#
MAP22S= 104416           252#
MODNAM  000000R          198#
MODSP   000224R          212     250#
MSGNS = 104403           252#
MSGSS = 104402           252#
MSGS  = 104401           252#
NULL  = 000000           252#    428
OPEN  = 000000           199     205     206     207     208     225     226     227     228     229     230     231     232
                         234     236     238     239     241     242     243     252#    474     475     476     477     478
                         479     480     481     482     483
OTOAS = 104420           252#
PASCNT  000034R          213#
PIRQS = 000004           252#    359
POPSP = 025726           252#
POPSP2= 022526           252#
PRTY  = 000000           203     252#
PRTY0 = 000000           252#
PRTY1 = 000040           252#
PRTY2 = 000100           252#
PRTY3 = 000140           252#
PRTY4 = 000200           252#
PRTY5 = 000240           252#
PRTY6 = 000300           252#
PRTY7 = 000340           252#
PS    = 177776           252#
PSW   = 177776           252#
PUSH  = 005746           252#
PUSH2 = 024646           252#
QERR1   001332R          435     438     454#
QERR2   001362R          430     446     461#
QPTR1   001434R          261*    352*    353*    354     356*    478#
QPTR2   001436R          262*    366     368*    369     371*    479#
RANDS = 104417           252#
RANNUM  000054R          221#
RDERR   001154R          384     388     423#
RDSERV  001220R          359     364#
REQ     001312R          263     264     450#    456     458     463     465
REQP1   001440R          263*    454*    455*    456     458*    480#
REQP2   001442R          264*    461     462*    463     465*    481#
RESTRT  000240R          240     258#
RES1    000056R          223#
RES2    000060R          224#
RINTR   000662R          297     299     301     303     305     307     309     311     313     315     317     319     321
                         323     325     327     352#
RQ      001272R          261     262     354     356     369     371     449#
RSER0   001030R          382     383     385#
RSER1   001044R          386     387     389#
RSER2   001062R          391     393#    447
RSTRT   000112R          240#
SBADR   000102R          233#    440*
SOPCNT  000042R          216#
SOPERS= 104406           252#
SOPPAS  000046R          218#
SPOINT  000032R          212#
```

```
SPSIZ = 000040                1#      245
SR1     000016R             205#
SR2     000020R             206#
SR3     000022R             207#
SR4     000024R             208#
START   000234R             211     255#
STAT    000476R             210#
STUP    000612R             240     334#
SVR0    000062R             225#
SVR1    000064R             226#
SVR2    000066R             227#
SVR3    000070R             228#
SVR4    000072R             229#
SVR5    000074R             230#
SVR6    000076R             231#
SYNFLG  001432R             335*    375     378*    477#
SYSCNT  000052R             220#
TRPDFD= 000022              252#
VECTOR  000010R             201#    272
WASADR  000104R             235#    437*
WDFR    000116R             242#
WDTO    000114R             241#    256*
XFLAG   000065R             199#
.       = 001450R           291#    292#    449#    450#

. ABS.  000000      000
        001450      001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 XPAAF0,XPAAF0/SOL/CRF:SYM=DD:COM,XPAAF0
 RUN-TIME:  1  1  .2 SECONDS
 RUN-TIME RATIO: 25/3=7.3
 CORE USED:  7K  (14 PAGES)
```