.REM >

IDENTIFICATION

PRODUCT CODE: AC-E715I-MC

PRODUCT NAME: CXDQAI0 DQ11 DEC/X11 MODULE

DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

1. ABSTRACT

DQA IS AN IOMOD THAT WILL EXERCISE UP TO EIGHT DQ11S. DURING A SINGLE PASS IT WILL DO 15 CHARACTER TRANSFERS OF SEQUENTIAL DATA AND CHECK THE STATUS REGISTERS AND SECONDARY REGISTERS. IT WILL TRANSFER A 256 EIGHT BIT BINARY COUNT PATTERN 128 TIMES FOR EACH END PASS RECORDED. ANY ERRORS DETECTED DURING THE PASS ARE REPORTED ON THE CONSOLE TTY.

2. REQUIREMENTS

HARDWARE: ONLY THE BASIC UNIT IS EXERCISED; SO A BASIC UNIT IS NEEDED. ANY EXTRA OPTIONS ON THE DQ11 WILL NOT BE EXERCISED.

STORAGE:: DQA REQUIRES:
1. DECIMAL WORDS: 768
2. OCTAL WORDS: 1400
3. OCTAL BYTES: 3000

3. PASS DEFINITION

ONE PASS OF THE DQA MODULE CONSISTS OF 24576. CYCLES OF AN INCREMENTAL DATA PATTERN TRANSFERRED AT 15. CHARACTER BURSTS.

4. EXECUTION TIME.

RUNNING ALONE ON AN 11/20 ONE PASS TAKES APPROX-IMATELY 20 SECONDS. NOTE: PASS TIME IS DEPENDENT UPON BAUD RATE.

5. CONFIGURATION REQUIREMENTS.

DEFAULT PARAMETERS:

DEVADR: 1 VECTOR: 1 BR1: 5 BR2: 5 DEVCNT: 1
USER MUST SPECIFY THE ADDRESS AND VECTOR OF THE FIRST DQ11 AT CONFIGURATION TIME.

6. DEVICE/OPTION SETUP

NO SPECIAL SET NECESSARY. (BASIC UNIT TESTED ONLY)

7. MODULE DESCRIPTION

A. TESTS FOR THE AVAILABILITY OF UP TO EIGHT DQ11'S
B. INITIALIZES ALL DQ11'S. SETS ACCORDINLY.
C. SETS ALL GO BITS AND LEAVES MODULE
D. GETS TRANS. INTERUPTS. GETS ALL RECV. INTERUPTS.
E. CHECKS ALL STATUS REGISTERS AND SECONDARY REGISTERS
F. CHECKS ALL DATA. REPORTS ANY ERROR FOUND.
G. PREPARES DATA TO TRANSMITTED AGAIN.
H. REPEATS A THROUGH G 128 X 256 CHARS.
J. REPORTS END PASS AND CONTINUES AS ABOVE.

8. OPERATOR OPTIONS

A. MODULE LOCATION DVID1 MAY BE CHANGED TO EXERCISE ANY COMBINATION OF DQ11S. BIT 0=DQ110 BIT 1=DQ11 1..... BIT 7=DQ11 7.

B. IF DVID1=0 AT RUN TIME NO DQ11S WILL BE EXERCISED.

9. NON STANDARD PRINTOUTS

NONE: ALL PRINTOUTS HAVE THE STANDARD FORMAT.

IF YOU NEED HELP IN RUNNING MODULE REFER TC DEC/X11 DOCUMENT.

>

```
                                        .LIST   SEQ,LOC,BIN
000000'                                 IOMOD   <DQAI >,1,1,5,5,,7100,31
000000'                                 MODULE  140000,DQAI ,1,1,5,5,,7100,31
                                        .TITLE  DQAI DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6      23-MAY-78
                                        .LIST   BIN
                                ;*****************************************************************************
000000'                         BEGIN:
000000' 050504  044501  040     MODNAM: .ASCII  /DQAI / ;MODULE NAME.
000005'     000                 XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 000001                  ADDR:   1+0             ;1ST DEVICE ADDR.
000010' 000001                  VECTOR: 1+0             ;1ST DEVICE VECTOR.
000012'     240                 BR1:    .BYTE   PRTY5+0         ;1ST BR LEVEL.
000013'     240                 BR2:    .BYTE   PRTY5+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' 007100                  ICONT:  7100                    ;# OF ITERATIONS PER PASS=7100
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' 000000                  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'                         SBADR:                          ;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                  ASD:    OPEN                    ;EXPECTED DATA.
000110' 000000                  AWAS:   OPEN                    ;ACTUAL DATA.
000112' 000332'                 RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114' 000000                  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' 000031                  IDNUM:  31                      ;MODULE IDENTIFICATION NUMBER=31
            000040                      .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
```



```
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224'                         MODSP:
                                ;*****************************************************************************
```

```
                                ;8 WORDS TO MEM PER ITERATION
                                ;8 WORDS FROM MEM PER ITERATION
                                ;2 INTERRUPTS PER ITERATION
                                ;SAVE DEVICE COUNT IN R0
                                ;BR IF AT LEAST ONE SELECTED
                                ;
                                ;KILL 1ST DEV-ALREADY COUNTED
                                ;LOOP TILL FOUND
                                ;SHIFT IN NEXT BIT
                                ;BR IF NO DEVICE HERE
                                ;DOUBLE WORDS
                                ;DOUBLE WORDS FROM
                                ;DOUBLE INTERRUPTS
                                ;ANY MORE DEVICES ?
                                ;BR IF YES
                                ;COPY THE DEVICE SELECTION PARAMETER
                                ;INTO SELECT
                                ;IF ZERO, NO DEVICES SELECTED-
                                ;DROP THE MODULE
                                ;DROP THE MODULE
                                ;ELIMINATE IRRELEVANT BITS


                                ;COPY SELECT INTO R1 FOR SETUP
                                ;IF SELECT WAS ZERO, GO DROP THE MODULE
                                ;IF SELECT WAS NOT ZERO, DO MODULE PROCESSING
                                ;LOAD R0 WITH FIRST VECTOR ADDRESS
                                ;LOAD R2 WITH FIRST DEVICE ADDRESS
                                ;POINT R3 TO BEGINNING OF JSR LINK TABLE
                                ;ISOLATE A LINE IN THE "C"BIT
                                ;IF SELECTED,GO SETUP ADDRESSES
                                ;IF NO MORE TO BE SELECTED,
                                ;GO SET UP BUFFERS,QUEUES,ETC.
                                ;IF MORE,UPDATE ALL REGISTERS..THE VECTOR
                                ;THE ADDRESS...
                                ;AND THE LINK TABLE POINTER
                                ;CHECK TO SEE IF OTHER DEVICES SELECTED
                                ;LOAD THE RECEIVE ROUTINE
                                ;POINTER IN THE PROPER VECTOR
                                ;LOAD THE RECEIVER BR LEVEL
                                ;UPDATE R0 TO NEXT VECTOR BOUNDARY
                                ;UPDATE R3 TO THE CSR INSERT LOCATION
                                ;LOAD THE RCV CSR INTO LINKING TABLE
                                ;UPDATE THE POINTER TO NEXT INSTRUCTION AND
                                ;POINT R2 TO THE TRANSMITTER CSR
                                ;LOAD THE TRANSMITTER ROUTINE
                                ;POINTER IN THE PROPER VECTOR
                                ;LOAD THE TRANSMIT BR LEVEL
                                ;UPDATE THE VECTOR POINTER
                                ;UPDATE THE LINK TABLE POINTER
                                ;LOAD THE TRANSMIT CSR INTO LINKING TABLE
                                ;UPDATE THE ADDRESS POINTER
                                ;GO SET UP NEXT DEVICE
```



```
                                ;SET UP TRANSMIT QUE ENTRY(IN)
                                ;POINTER
                                ;SET UP TRANSMIT QUE RETRIEVAL
                                ;(OUT) POINTER
                                ;SET UP RECEIVER QUE POINTERS
                                ;
                                ;SET UP ERROR QUE POINTERS
                                ;
                                ;LOAD R4 WITH NUMBER OF BUFFER
                                ;LOCATIONS TO BE CLEARED
                                ;BEGIN CLEARING BUFFERS AT THE
                                ;TRANSMIT BUFFER
                                ;ZERO EACH LOCATION AND POINT R3 TO
                                ;NEXT WORD
                                ;HAVE ALL LOCATIONS BEEN CLEARED?
                                ;IF NO, CLEAR THE NEXT ONE


                                ;LOAD DEVICE SELECTION PARAMETER
                                ;SETUP TRANSFER COMPLETION FLAG
                                ;COPY THE BASE ADDRESS
                                ;GET THE PHYSICAL ADDRESS OF FIRST
                                ;RECFIVER BUFFER
                                ;GET PHYSICAL ADDRESS FROM 16-BIT VA
                                ;SAVE RECEIVE BUFFER PHYSICAL ADDRESS
                                ;GET THE PHYSICAL ADDRESS OF TRANSMITTER BUFFER
                                ;GET PHYSICAL ADDRESS FROM 16-BIT VA
                                ;SAVE TRANSMITTER BUFFER PHYSICAL ADDRESS
                                ;BE SURE CARRY BIT IS CLEAR BEFORE ROTATING BITS
                                ;ALIGN THE EXTENDED ADDRESS BITS IN ORDER
                                ;TO SET BITS 13 AND 14 OF THE REG/ERR
                                ;REGISTER
                                ;ISOLATE DEVICE SELECTION FLAG IN "C" BIT
                                ;IF SELECTED, GO INITIATE TRANSFER
                                ;IF NO MORE SELECTED, GO SETUP DATA
                                ;UPDATE POINTER TO NEXT DEVICE ADDRESS
                                ;UPDATE POINTER TO NEXT RECEIVER BUFFER
                                ;PROCESS NEXT DEVICE
                                ;CLEAR SECONDARY REGISTERS POINTER
                                ;FROM REG/ERR REGISTER(BITS 8-11)
                                ;SET COUNT TO 16. FOR SECONDARY REGISTER
                                ;CLEARING
                                ;ENABLE EXTENDED BITS WRITING(REG/ERR BIT 12)
                                ;CLEAR BITS 16 AND 17 OF THE
                                ;SECONDARY REGISTER BY CLEARING 13
                                ;AND 14 OF THE ERR/REG REGISTER
                                ;CLEAR A SECONDARY REGISTER
                                ;POINT TO NEXT SECONDARY REGISTER
                                ;REDUCE COUNT. ARE ALL SIXTEEN DONE?
                                ;IF NO,GO DO NEXT ONE
                                ;SET UP COUNTER TO CLEAR CHAR. DETECT REGS.
                                ;IF YES, CLEAR CHARACTER DETECT REGISTERS
                                ;IS -BB OPTION HERE ?
                                ;NO DONT CLEAR IT,CLEARING IT
                                ;WIPES OUT HIGH ORDER BITS.
                                ;SELECT SECONDARY REGISTER 10
```

```
  302  000732' 112761  000034  000005          MOVB    #34,5(R1)               ;SELECT SECONDARY REG. 14
  303  000740' 005061  000006                  CLR     6(R1)
  304  000744' 005302                          DEC     R2                      ;ARE ALL REGISTERS CLEAR?
  305  000746' 001403                          BEQ     7$
  306  000750' 105361  000001                  DECB    1(R1)                   ;SELECT THE NEXT PAIR OF CHARACTER DETECT REGIST
  307  000754' 003361                          BGT     5$                      ;CLEAR 16. PAIRS OF REGISTERS
  308                                                                          ;HIGH BYTE OF RECEIVER CSR
  309  000756' 112761  000012  000005  7$:    MOVB    #12,5(R1)               ;SELECT SECONDARY REGISTER 12
  310                                                                          ;(MISCELLANEOUS REGISTER)
  311  000764' 052761  000040  000006          BIS     #BIT5,6(R1)             ;MASTER CLEAR THE DQ11 BY
  312                                                                          ;SETTING BIT 5 OF THE MISC. REGISTER
  313  000772' 112761  000020  000005          MOVB    #20,5(R1)               ;ENABLE EXTENDED MEMORY BITS
  314                                                                          ; AND SELECT THE RECEIVER BUSS
  315                                                                          ;ADDRESS REGISTER (SEC. REG.0)
  316  001000' 156761  001770  000005          BISB    EA,5(R1)                ;SET THE EA BITS
  317  001006' 016761  001746  000006          MOV     RCVADR,6(R1)            ;LOAD ADDRESS OF RECEIVER BUFFER
  318  001014' 105261  000005                  INCB    5(R1)                   ;SELECT THE RECEIVER BYTE COUNT REGISTER
  319                                                                          ;(SECONDARY REG. 1)
  320  001020' 012761  177761  000006          MOV     #-15.,6(R1)             ;SET COUNT FOR 15 CHARACTERS
  321  001026' 112761  000022  000005          MOVB    #22,5(R1)               ;SELECT THE TRANSMITTER BUSS ADDRESS REGISTER
  322  001034' 156761  001734  000005          BISB    EA,5(R1)                ;SET THE EA BITS
  323                                                                          ;(SECONDARY REGISTER 2)
  324  001042' 016761  001714  000006          MOV     XMTADR,6(R1)            ;LOAD ADDRESS OF TRANSMITTER BUFFER
  325  001050' 105261  000005                  INCB    5(R1)                   ;SELECT THE TRANSMITTER BYTE COUNT REGISTER
  326                                                                          ;(SECONDARY REGISTER 3)
  327  001054' 012761  177755  000006          MOV     #-19.,6(R1)             ;LOAD TRANSMISSION BYTE COUNT
  328  001062' 112761  000031  000005          MOVB    #31,5(R1)               ;SELECT THE SYNC CHARACTER REGISTER
  329                                                                          ;(SECONDARY REGISTER 11)
  330  001070' 016761  001326  000006          MOV     SYNC,6(R1)              ;LOAD THE SYNC REGISTER WITH
  331                                                                          ;TWO COPIES OF THE SYNC CHARACTER
  332  001076' 105261  000005                  INCB    5(R1)                   ;SELECT THE MISC. REGISTER (REG. 12)
  333  001102' 012761  004010  000006          MOV     #4010,6(R1)             ;ENABLE TEST LOOP(BIT 3) AND
  334                                                                          ;SELECT 9 BITS/CHARACTER (1000 IN BITS 8-11)
  335
  336  001110' 052711  000042                  BIS     #42,(R1)                ;ENABLE RECEIVER INTERRUPT(BIT 5)
  337                                                                          ;AND STRIP SYNC(BIT 1)
  338  001114' 000642                          BR      2$                      ;ADJUST POINTERS FOR NEXT DEVICE
  339
  340
  341  001116' 012701  002426'          INITIAL:MOV     #XMTBUF,R1              ;POINT R1 TO BEGINNING OF TRANSMIT TEXT
  342  001122' 012702  000017                  MOV     #15.,R2                 ;USE R2 AS A LOOP COUNTER
  343  001126' 122767  000026  001643  1$:     CMPB    #26,DATA                ;IS THE DATUM THE SYNC CHARACTER?
  344  001134' 001002                          BNE     5$                      ;IF NO,GO MOVE IT INTO THE TRANSMITTER BUFFER
  345  001136' 105267  001635                  INCB    DATA                    ;IF YES, SKIP IT.
  346  001142' 116721  001631          5$:     MOVB    DATA,(R1)+              ;USING DATA,BUILD A BINARY TEST PATTERN
  347  001146' 105267  001625                  INCB    DATA                    ;SET DATA TO NEXT CHARACTER
  348  001152' 005302                          DEC     R2                      ;REDUCE COUNT. HAVE 15 CHARACTERS BEEN MADE?
  349  001154' 001364                          BNE     1$                      ;IF NO, CONTINUE BUILDING TEXT .
  350  001156' 016700  001574                  MOV     SELECT,R0               ;IF YES,BEGIN TRANSFERS
  351                                                                          ;LOAD DEVICE SELECTION PARAMETER INTO R0
  352  001162' 016701  176620                  MOV     ADDR,R1                 ;LOAD BASE ADDRESS INTO R1
  353  001166' 006200                  2$:     ASR     R0                      ;ISOLATE A LINE SELECTION FLAG IN "C" BIT
  354  001170' 103404                          BCS     4$                      ;IF SELECTED GO START DEVICE
  355  001172' 001410                          BEQ     TMRSET                  ;IF NO MORE SELECTED GO START WATCHDOG TIMER
  356  001174' 062701  000010          3$:     ADD     #10,R1                  ;UPDATE ADDRESS POINTER
  357  001200' 000772                          BR      2$                      ;GO START NEXT DEVICE
```



```
  358  001202' 005211                  4$:     INC     (R1)                    ;START RECEIVER (BIT 0=GO BIT)
  359  001204' 012761  000051  000002          MOV     #51,2(R1)               ;ENABLE TRANSMITTER(BIT 0) AND ERROR INTER-
  360                                                                          ;RUPT(BIT 3) ANDTRANSMITTER INTER-
  361                                                                          ;RUPT(BIT 5)
  362  001212' 000770                          BR      3$                      ;GO ADJUST POINTER TO NEXT DEVICE
  363
  364
  365  001214' 012767  000005  001544  TMRSET: MOV     #5,TMRCNT               ;LOAD THE TIMER COUNTING FACTOR
  366  001222' 005004          TIMER:  CLR     R4                              ;USING R4,RETURN TO MONITOR 65536 TIMES
  367  001224'                 3$:
  368  001224' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
  369  001230' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
  370  001234' 105767  001536                  TSTB    DONFLG                  ;IF DONFLG IS CLEAR,EACH SELECTED DEVICE WAS
  371                                                                          ;SUCCESSFUL
  372  001240' 001433                          BEQ     FINISH                  ;IF SO PERFORM ENDPASS HOUSEKEEPING
  373  001242' 005304                          DEC     R4                      ;IF NOT, REDUCE COUNT AND BREAK AGAIN
  374  001244' 001367                          BNE     3$                      ;BREAK IF COUNT NOT EXCEEDED
  375  001246' 005367  001514                  DEC     TMRCNT          ;REDUCE TIMING FACTOR
  376  001252' 001363                          BNE     TIMER                   ;BREAK AGAIN IF NO TIMEOUT
  377  001254' 116703  001516                  MOVB    DONFLG,R3               ;IF TIMEOUT OCCURRED,SAVE PRESENT FLAGS
  378                                                                          ; IN R3
  379  001260' 040367  001472                  BIC     R3,SELECT               ;USE R3 TO DEACTIVATE HUNG DEVICE
  380                                                                          ;BY CLEARING ACTIVE SELECTION FLAG FROM
  381                                                                          ;DEVICE SELECTION PARAMETER
  382  001264' 006003                  1$:     ROR     R3                      ;DETERMINE WHICH LINE WAS
  383                                                                          ;BAD FOR REPORTING PURPOSES
  384  001266' 103402                          BCS     2$                      ;IF THIS IS THE LINE,R4 CONTAINS CORRECT
  385                                                                          ;LINE NUMBER...GO REPORT IT
  386  001270' 005204                          INC     R4                      ;IF NOT, INCREMENT R4, WHICH WAS INITIALLY
  387                                                                          ;0 FROM THE PREVIOUS LOOP
  388  001272' 000774                          BR      1$
  389  001274' 010467  001104          2$:     MOV     R4,NUMBA1               ;SAVE R4
  390                                          ;*****************************************************************
  391                                                                  ;CONVERT NUMBA1 TO ASCII AND
  392                                                                  ;STORE AT M1
  393  001300' 104420  000000' 002404'         OTOA$,BEGIN,NUMBA1,M1
  394  001306' 002367'                         ;*****************************************************************
  395
  396  001310' 104403  000000' 002354'         MSGN$,BEGIN,HUNG        ;ASCII MESSAGE CALL WITH COMMON HEADER
  397  001316' 005767  001434                  TST     SELECT                  ;ARE THERE ANY DRIVES REMAINING?
  398  001322' 001002                          BNE     FINISH                  ;IF YES,REDUCE COUNT AND CONTINUE
  399                                                                          ;IF NOT,DROP THE MODULE
  400  001324' 104410  000000'                 END$,BEGIN              ;
  401
  402
  403  001330'                         FINISH:
  404  001330' 104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
  405                                                                  ;MONITOR SHALL TEST END OF PASS
  406  001334' 000167  177104                  JMP     SETUP2                  ;START NEXT TRANSFER
  407
  408
  409
  410
  411
  412
  413  001340' 010577  001042          XMTINT: MOV     R5,@XMTQPI              ;LOAD THE OFFSET TO THE CSR INTO TRANSMITTER QUE
```

```
001344' 062767  000002  001034          ADD     #2,XMTQPI               ;UPDATE THE QUEUE ENTRY POINTER
001352' 022767  002666' 001026          CMP     #XMTQUE+20,XMTQPI       ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001360' 001003                          BNE     1$                      ;IF IT HAS NOT EXCEEDED THE BOUNDARY,
                                                                        ;GO DEFER SERVICE TO LEVEL 0
001362' 012767  002646' 001016          MOV     #XMTQUE,XMTQPI          ;IF IT HAS, UPDATE THE POINTER
001370' 012605                  1$:     MOV     (SP)+,R5                ;RESTORE THE PREVIOUS R5 VALUE FROM STACK
                                        ;-----------------------------------------------------------------
001372' 000004  000000' 001400'         $IRQS,BEGIN,XMTSRV              ; QUEUE UP TO CONTINUE AT XMTSRV AND RTI
                                        ;-----------------------------------------------------------------
001400' 017701  001004          XMTSRV: MOV     @XMTQPO,R1              ;FETCH THE OFFSET FROM THE QUEUE
001404' 062767  000002  000776          ADD     #2,XMTQPO               ;UPDATE THE QUEUE RETRIEVAL POINTER
001412' 022767  002666' 000770          CMP     #XMTQUE +20,XMTQPO      ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001420' 001003                          BNE     1$                      ;IF NO, GO PROCESS INTERRUPT
001422' 012767  002646' 000760          MOV     #XMTQUE,XMTQPO          ;IF YES, RESET POINTER TO BEGINNING OF QUEUE
001430' 011100                  1$:     MOV     (R1),R0                 ;LOAD THE CSR ADDRESS INTO R0
001432' 005760  000002                  TST     2(R0)                   ;IF THERE WAS AN ERROR, BIT 15 OF REG/ERR IS SET
001436' 002017                          BGE     2$                      ;IF NO ERROR, PROCESS THE INTERRUPT
001440' 010067  176434                  MOV     R0,CSRA                 ;ON ERROR, LOAD THE DEVICE ADDRESS INTO CSRA
001444' 011067  176432                  MOV     (R0),ACSR               ;LOAD THE CONTENTS INTO ACSR
001450' 016067  000002  176426          MOV     2(R0),ASTAT             ;LOAD THE REGISTER/ERROR REG. INTO STATC
001456' 005067  176424                  CLR     ERRTYP          ;UNKNOWN ERROR
                                        ;*****************************************************************
001462' 104405  000000' 000000          $RDERS,BEGIN,NULL       ;ERROR FLAG SET-CONTENTS OF REG.ERR IN STATC
                                        ;*****************************************************************
001470' 005004                          CLR     R4                      ;COMPENSATE FOR PRINT DELAY ON TIMER
001472' 105067  001300                  CLRB    DONFLG
001476' 042710  000001          2$:     BIC     #BIT0,(R0)              ;DISABLE TRANSMITTER
001502' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERPUPT.


001506' 010577  000700          RCVINT: MOV     R5,@RCVQPI              ;ENTER OFFSET INTO RECEIVER QUEUE
001512' 062767  000002  000672          ADD     #2,RCVQPI               ;UPDATE THE RECEIVER ENTRY POINTER
001520' 022767  002706' 000664          CMP     #RCVQUE+20,RCVQPI       ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001526' 001003                          BNE     1$                      ;IF NOT,CONTINUE PROCESSING
001530' 012767  002666' 000654          MOV     #RCVQUE,RCVQPI          ;IF SO,POINT POINTER TO QUEUE BEGINNING
001536' 012605                  1$:     MOV     (SP)+,R5                ;RESTORE PREVIOUS VALUE OF R5
                                        ;-----------------------------------------------------------------
001540' 000004  000000' 001546'         $IRQS,BEGIN,RCVSRV              ; QUEUE UP TO CONTINUE AT RCVSRV AND RTI
                                        ;-----------------------------------------------------------------
001546' 017700  000642          RCVSRV: MOV     @RCVQPO,R0              ;RETRIEVE OFFSET FROM RECEIVER QUEUE
001552' 062767  000002  000634          ADD     #2,RCVQPO               ;UPDATE THE QUEUE POINTER
001560' 022767  002706' 000626          CMP     #RCVQUE+20,RCVQPO       ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001566' 001003                          BNE     1$                      ;IF NOT,CONTINUE PROCESSING
001570' 012767  002666' 000616          MOV     #RCVQUE,RCVQPO          ;IF SO,POINT POINTER TO  QUEUE BEGINNING
001576' 011001                  1$:     MOV     (R0),R1                 ;PLACE CSR ADDRESS IN R1
001600' 005761  000004                  TST     4(R1)                   ;WERE THERE ANY ERRORS(BIT 15 OF REG/ERR REG)?
001604' 002014                          BGE     2$                      ;IF NOT,CONTINUE PROCESSING
001606' 010167  176266                  MOV     R1,CSRA                 ;IF SO,LOAD BASE ADDRESS INTO CSRA
001612' 011167  176264                  MOV     (R1),ACSR               ;MOVE CONTENTS OF CSR TO  ACSR
001616' 016167  000004  176260          MOV     4(R1),ASTAT             ;LOAD REG/ERR REG.  CONTENTS INTO STATC
001624' 005067  176256                  CLR     ERRTYP          ;UNKNOWN ERROR
                                        ;*****************************************************************
001630' 104405  000000' 000000          $RDERS,BEGIN,NULL       ;DQ11 ERROR FLAG SET-REG/ERR IN STATC
                                        ;*****************************************************************
001636' 122711  000246          2$:     CMPB    #246,(R1)               ;
```



```
001642' 001412                          BEQ     3$
001644' 010167  176230                  MOV     R1,CSRA
001650' 011167  176226                  MOV     (R1),ACSR
001654' 012767  000017  176224          MOV     #17,ERRTYP              ;UNKNOWN RECEIVER ERROR
                                        ;*****************************************************************
001662' 104405  000000' 000000          $RDERS,BEGIN,NULL       ;RECEIVER STATUS ERROR
                                        ;*****************************************************************
001670' 122761  000254  000002  3$:     CMPB    #254,2(R1)              ;
001676' 001416                          BEQ     CKDATA                  ;IF NO ERRORS,BEGIN COMPARING DATA
001700' 011167  000002  176174          MOV     2(R1),ACSR              ;LOAD TRANSMITTER STATUS INTO ACSR
001706' 010167  176166                  MOV     R1,CSRA                 ;LOAD STATUS REGISTER ADDRESS
001712' 062767  000002  176160          ADD     #2,CSRA                 ;ALTER BASE ADDRESS TO SHOW TRANSMITTER ADDRESS
001720' 012767  000020  176160          MOV     #20,ERRTYP              ;UNKNOWN XMITTER ERROR
                                        ;*****************************************************************
001726' 104405  000000' 000000          $RDERS,BEGIN,NULL       ;TRANSMITTER STATUS ERROR
                                        ;*****************************************************************


001734' 012767  002446' 001016  CKDATA: MOV     #RCVBF0,RCVADR          ;RESTORE BASE VALUE OF RECEIVER BUFFERS
001742' 012767  000017  001014          MOV     #15.,DATACT             ;LOAD THE NUMBER OF CHARACTERS TO BE CHECKED
001750' 016002  000002                  MOV     2(R0),R2                ;LOAD THE LINE NUMBER OF THIS DQ11
001754' 001405                  1$:     BEQ     2$                      ;WHEN  R2  IS  0,THE CORRECT RECEIVER BUFFER
                                                                        ;HAS BEEN SELECTED
001756' 062767  000020  000774          ADD     #20,RCVADR              ;IF R2 IS NOT 0,POINT RCVADR TO NEXT BUFFER
001764' 005302                          DEC     R2                      ;REDUCE  THE LINE NUMBER BY 1
001766' 000772                          BR      1$                      ;GO SEE IF THE CORRECT BUFFER HAS BEEN DETERMINE
001770' 016702  000764          2$:     MOV     RCVADR,R2               ;LOAD R2 WITH THE START OF THE RECEIVER BUFFER
001774' 012767  002446' 000756          MOV     #RCVBF0,RCVADR          ;RESTORE BASE VALUE OF RECEIVER BUFFERS
002002' 012703  002426'                 MOV     #XMTBUF,R3              ;LOAD  R3  WITH  THE  START  OF  TRANSMITTER TEXT
002006' 122223                  3$:     CMPB    (R2)+,(R3)+             ;MATCH CHARACTERS. ARE THEY THE SAME?
002010' 001453                          BEQ     4$                      ;IF YES,  GO  PROCESS  NEXT  CHARACTER
002012' 010167  176062                  MOV     R1,CSRA                 ;LOAD THE BASE ADDRESS INTO CSRA
002016' 011167  176060                  MOV     (R1),ACSR               ;LOAD RECEIVER CSR CONTENTS INTO ACSR
002022' 114267  176062                  MOVB    -(R2),AWAS              ;LOAD THE ACTUAL RECEIVED CHARACTER
002026' 114367  176054                  MOVB    -(R3),ASB               ;LOAD THE TRANSMITTED CHARACTER
002032' 010267  176046                  MOV     R2,WASADR               ;LOAD THE CHARACTER RECEIVED ADDRESS
002036' 010367  176040                  MOV     R3,SBADR                ;LOAD THE CHARACTER TRANSMITTED ADDRESS
002042' 016705  000350                  MOV     ERRQPI,R5               ;USING R5,LOAD THE PRESENT VALUES OF THE REGISTE
                                                                        ;INTO THE ERROR QUEUE, UNQUEUEING THEM AFTER THE
                                                                        ;DATA ERROR CALL. THIS ASSURES THAT THE REGISTER
                                                                        ;CONTENTS ARE VALID
002046' 010025                          MOV     R0,(R5)+                ;SAVE THE QUEUE OFFSET
002050' 010125                          MOV     R1,(R5)+                ;SAVE THE CSR ADDRESS
002052' 010225                          MOV     R2,(R5)+                ;SAVE THE PRESENT RECEIVE BUFFER POINTER
002054' 010325                          MOV     R3,(R5)+                ;SAVE THE CURRENT TRANSMIT BUFFER POINTER
002056' 010425                          MOV     R4,(R5)+                ;SAVE THE TIMER COUNTDOWN
002060' 020527  002732'                 CMP     R5,#ERRQUE+20.          ;HAS THE POINTER EXCEEDED THE QUEUE BOUNDARY?
002064' 103402                          BLO     5$                      ;IF SO,DO NOT RESET POINTER
002066' 012705  002706'                 MOV     #ERRQUE,R5              ;POINT POINTER TO BEGINNING OF QUEUE
002072' 010567  000320          5$:     MOV     R5,ERRQPI               ;RESTORE ERROR QUE ENTRY POINTER
                                        ;*****************************************************************
002076' 104404  000000'                 DATER$,BEGIN            ;DATA ERROR!!!
                                        ;*****************************************************************
002102' 016705  000312                  MOV     ERRQPO,R5               ;LOAD R5 WITH ERROR QUEUE RETRIEVAL POINTER
                                                                        ;RETRIEVE,USING R5, THE FOLLOWING FROM THE ERROR
```

```
                                                ;QUEUE,PLACING THEM IN THE CORRESPONDING REGISTE
002106' 012500                  MOV     (R5)+,R0        ;THE OFFSET....
002110' 012501                  MOV     (R5)+,R1        ;THE BASE ADDRESS....
002112' 012502                  MOV     (R5)+,R2        ;THE CURRENT RECEIVER BUFFER POINTER....
002114' 012503                  MOV     (R5)+,R3        ;THE CURRENT TRANSMITTER BUFFER POINTER....
002116' 012504                  MOV     (R5)+,R4        ;AND THE CORRECT TIMER VALUE
002120' 020527  002732'         CMP     R5,#ERRQUE+20.  ;HAS THE POINTER EXCEEDED THE QUEUE BOUNDARY?
002124' 103402                  BLO     6$              ;IF NOT,DO NOT RESET IT
002126' 012705  002706'         MOV     #ERRQUE,R5      ;IF SO, RESET THE POINTER TO QUEUE BEGINNING
002132' 010567  000262  6$:     MOV     R5,ERRQPO       ;RESTORE THE RETREIVAL POINTER
002136' 122223          4$:     CMPB    (R2)+,(R3)+     ;AUTOINCREMENT THE BUFFER POINTERS TO
                                                        ;POINT TO THE CORRECT CHARACTER
002140' 005367  000620          DEC     DATACT          ;REDUCE CHARACTER COUNT.ARE ALL 15 DONE?
002144' 003320                  BGT     3$              ;IF NO, DO NEXT CHARACTER


002146' 005003          RCVDON: CLR     R3              ;IF YES,SETUP R3 TO TURN OFF FLAG IN DONFLG
                                                        ;FOR THIS LINE
002150' 016002  000002          MOV     2(R0),R2        ;LOAD THE LINE NUMBER INTO R2
002154' 000261                  SEC                     ;USING THE CARRY BIT, CREATE A ONE-BIT MASK
002156' 006103          1$:     ROL     R3              ;POINT THE MASK TO THE NEXT LINE
002160' 005302                  DEC     R2              ;REDUCE THE LINE NUMBER
002162' 002375                  BGE     1$              ;IF POSITIVE OR 0, GO SHIFT BIT AGAIN
002164' 140367  000606          BICB    R3,DONFLG       ;IF NEGATIVE, THE MASK IS CORRECTLY ALIGNED-
                                                        ;CLEAR THE DONE BIT FOR THIS LINE
002170' 104400  000000'         EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

002174' 004567  177306  LNKTAB: JSR     R5,RCVINT       ;LINK FOR RECEIVER 0
002200' 000000                  0
002202' 000000                  0
002204' 004567  177130          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 0
002210' 000000                  0
002212' 004567  177270          JSR     R5,RCVINT       ;LINK FOR RECEIVER 1
002216' 000000                  0
002220' 000001                  1
002222' 004567  177112          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 1
002226' 000000                  0
002230' 004567  177252          JSR     R5,RCVINT       ;LINK FOR RECEIVER 2
002234' 000000                  0
002236' 000002                  2
002240' 004567  177074          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 2
002244' 000000                  0
002246' 004567  177234          JSR     R5,RCVINT       ;LINK FOR RECEIVER 3
002252' 000000                  0
002254' 000003                  3
002256' 004567  177056          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 3
002262' 000000                  0
002264' 004567  177216          JSR     R5,RCVINT       ;LINK FOR RECEIVER 4
002270' 000000                  0
002272' 000004                  4
002274' 004567  177040          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 4
002300' 000000                  0
002302' 004567  177200          JSR     R5,RCVINT       ;LINK FOR RECEIVER 5
002306' 000000                  0
002310' 000005                  5
```



```
002312' 004567  177022          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 5
002316' 000000                  0
002320' 004567  177162          JSR     R5,RCVINT       ;LINK FOR RECEIVER 6
002324' 000000                  0
002326' 000006                  6
002330' 004567  177004          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 6
002334' 000000                  0
002336' 004567  177144          JSR     R5,RCVINT       ;LINK FOR RECEIVER 7
002342' 000000                  0
002344' 000007                  7
002346' 004567  176766          JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 7
002352' 000000                  0

002354' 002360'         HUNG:   MESSAG
002356' 177777                  -1
002360' 042504  044526  042503  MESSAG: .ASCII  'DEVICE '
002366'     040
002367' 000006          M1:     .BLKB   6
002375'     040  052510  043516 M2:     .ASCIZ  ' HUNG'
002402'     000
        002404'                 .EVEN

002404' 000000          NUMBA1: .WORD   0
002406' 000000          XMTQPI: .WORD   OPEN            ;POINTER FOR TRANSMITTER QUEUE INPUT
002410' 000000          XMTQPO: .WORD   OPEN            ;POINTER FOR TRANSMITTER QUEUE OUTPUT
002412' 000000          RCVQPI: .WORD   OPEN            ;POINTER FOR RECEIVER QUEUE INPUT
002414' 000000          RCVQPO: .WORD   OPEN            ;POINTER FOR RECEIVER QUEUE OUTPUT
002416' 000000          ERRQPI: .WORD   OPEN            ;POINTER FOR ERROR QUEUE INPUT
002420' 000000          ERRQPO: .WORD   OPEN            ;POINTER FOR ERROR QUEUE OUTPUT

002422'     026     026 SYNC:   .BYTE   26,26
002424'     026     026         .BYTE   26,26
002426' 000010          XMTBUF: .BLKW   8.              ;TRANSMITTER CHARACTER BUFFER
002446' 000010          RCVBF0: .BLKW   8.              ;DEVICE 0 RECEIVER BUFFER
002466' 000010          RCVBF1: .BLKW   8.              ;DEVICE 1 RECEIVER BUFFER
002506' 000010          RCVBF2: .BLKW   8.              ;DEVICE 2 RECEIVER BUFFER
002526' 000010          RCVBF3: .BLKW   8.              ;DEVICE 3 RECEIVER BUFFER
002546' 000010          RCVBF4: .BLKW   8.              ;DEVICE 4 RECEIVER BUFFER
002566' 000010          RCVBF5: .BLKW   8.              ;DEVICE 5 RECEIVER BUFFER
002606' 000010          RCVBF6: .BLKW   8.              ;DEVICE 6 RECEIVER BUFFER
002626' 000010          RCVBF7: .BLKW   8.              ;DEVICE 7 RECEIVER BUFFER


002646' 000010          XMTQUE: .BLKW   8.              ;TRANSMITTER SERVICE QUEUE
002666' 000010          RCVQUE: .BLKW   8.              ;RECEIVER SERVICE QUEUE
002706' 000024          ERRQUE: .BLKW   20.             ;ERROR SERVICE QUEUE


002756' 000000          SELECT: OPEN                    ;DEVICE SELECTION PARAMETER
002760' 000000          RCVADR: OPEN                    ;PHYSICAL ADDRESS OF RECEIVER BUFFERS
002762' 000000          XMTADR: OPEN                    ;PHYSICAL ADDRESS OF TRANSMITTER BUFFER
                                                        ;FOR ALL DEVICES
002764' 000000          DATACT: .WORD   OPEN            ;COUNTER FOR NUMBER OF DATA ITEMS
002766' 000000          TMRCNT: .WORD   OPEN            ;LOCATION FOR WATCHDOG TIMER FACTOR
002770' 000000          VA:     .WORD   OPEN            ;LOCATION OF VIRTUAL ADDRESS PARAMETER
002772' 000000          PA:     .WORD   OPEN            ;LOCATION OF PHYSICAL ADDRESS PARAMETER
```

```
   638  002774' 000000          EA:     .WORD   OPEN            ;LOCATION OF EXTENDED ADDRESS BITS PARAMETER
   639  002776' 000             DONFLG: .BYTE   OPEN            ;TRANSFER COMPLETION FLAG
   640  002777' 000             DATA:   .BYTE                   ;USED TO BUILD TEST PATTERN
   641                          .EVEN
   642          000001          .END
```



```
ACSR    000102R          172#    432*    463*    472*    479*    503*
ACTVAT  000526R          263#
ADDR    000006R          138#    218     265     352
ADDR22= 001000           190#
ASB     000106R          176#    505*
ASTAT   000104R          174#    433*    464*
AWAS    000110R          177#    504*
BEGIN   000000R          135#    195     210     268     271     368     369     393     396     400     404     421     476
                         441     452     467     475     484     522     552
BIT0    = 000001         190#    440
BIT1    = 000002         190#
BIT10   = 002000         190#
BIT11   = 004000         190#
BIT12   = 010000         190#    287
BIT13   = 020000         190#
BIT14   = 040000         190#
BIT15   = 100000         190#
BIT2    = 000004         190#
BIT3    = 000010         190#
BIT4    = 000020         190#
BIT5    = 000040         190#    311
BIT6    = 000100         190#
BIT7    = 000200         190#
BIT8    = 000400         190#
BIT9    = 001000         190#
BREAK$= 104407           190#    368     369
BR1     000012R          140#    238
BR2     000013R          141#    230
BTOD$   = 104421         190#
CDATA$= 104412           190#
CKDATA  001734R          478     489#
CONFIG  000056R          160#
CSRA    000100R          170#    431*    462*    471*    480*    481*    502*
DATA    002777R          343     345*    346     347*    640#
DATACT  002764R          490*    538*    634#
DATCK$= 104411           190#
DATER$= 104404           190#    522
DONFLG  002776R          264*    370     377     439*    550*    639#
DROP    000326R          209#    215
DVID1   000014R          142#    193     205
EA      002774R          274#    316     322     638#
ENDIT$= 104413           190#    404
END$    = 104410         190#    195     210     400
ERRQPI  002416R          252*    508     520*    609#
ERRQPO  002420R          253*    524     535*    610#
ERRQUE  002706R          252     253     517     519     532     534     627#
ERRTYP  001106R          175#    434*    465*    473*    482*
EXIT$ = 104400           190#    441     552
FINISH  001330R          372     398     403#
GETPA$= 104415           190#    268     271
GWBUF$= 104414           190#
HRDCNT  000044R          155#
HRDER$= 104405           190#    436     467     475     484
HRDPAS  000050R          157#
HUNG    002354R          396     595#
ICONT   000036R          152#
```

```
ICOUNT  000040R          153#
IDNUM   000122R          182#
INIT    000030R          149#
INITIA  001116R          279     341#
INTR    000120R          181#    192*    202*
LNKTAB  002174R          219     554#
MAP22S= 104416           190#
MESSAG  002360R          595     597#
MODNAM  000000R          136#
MODSP   000224R          150     188#
MSGN$ = 104403           190#    396
MSGS$ = 104402           190#
MSG$  = 104401           190#
M1      002367R          593     599#
M2      002375R          600#
NULL  = 000000           190#    436     467     475     484
NUMBA1  002404R          389*    393     604#
OPEN  = 000000           137     143     144     145     146     163     164     165     166     167     168     169     170
                         172     174     176     177     179     180     181     190#    605     606     607     608     609
                         610     630     631     632     634     635     636     637     638     639
OTOA$ = 104420           190#    393
PA      002772R          269     272     637#
PASCNT  000034R          151#
PIRQ$ = 000004           190#    421     452
POPSP = 005726           190#
POPSP2= 022626           190#
PRTV  = 000000           190#
PRTV0 = 000000           190#
PRTV1 = 000040           190#
PRTV2 = 000100           190#
PRTV3 = 000140           190#
PRTV4 = 000200           190#
PRTV5 = 000240           140     141     190#
PRTV6 = 000300           190#
PRTV7 = 000340           190#
PS    = 177776           190#
PSW   = 177776           190#
PUSH  = 005746           190#
PUSH2 = 024646           190#
RAND$ = 104417           190#
RANNUM  000054R          159#
RCVADR  002760R          269*    281*    317     489*    494*    497     498*    631#
RCVBF0  002446R          266     489     498     615#
RCVBF1  002466R          616#
RCVBF2  002506R          617#
RCVBF3  002526R          618#
RCVBF4  002546R          619#
RCVBF5  002566R          620#
RCVBF6  002606R          621#
RCVBF7  002626R          622#
RCVDON  002146R          143#
RCVINT  001506R          445#    554     559     564     569     574     579     584     589
RCVQPI  002412R          250*    445*    446*    447     449*    607#
RCVQPO  002414R          251*    454     455*    456     458*    608#
RCVQUE  002666R          250     251     447     449     456     458     626#
RCVSRV  001546R          452     454#
```



```
RESTRT  000332R          178     207     211#
RES1    000056R          161#
RES2    000060R          162#
RSTRT   000112R          178#
SRADR   000102R          171#    507*
SELECT  002756R          205*    211*    214     263     350     379*    397     630#
SETUP1  000336R          214#
SETUP2  000444R          222     246#    406
SOFCNT  000042R          154#
SOFERR= 104406           190#
SOFPAS  000046R          156#
SPOINT  000032R          150#
SPSIZ = 000040             1#    183
SR1     000016R          143#
SR2     000020R          144#
SR3     000022R          145#
SR4     000024R          146#
START   002224R          149     190#
STAT    000026R          148#
SVR0    000062R          163#
SVR1    000064R          164#
SVR2    000066R          165#
SVR3    000070R          166#
SVR4    000072R          167#
SVR5    000074R          168#
SVR6    000076R          169#
SYNC    002422R          270     330     612#
SYSCNT  000052R          158#
TIMER   001222R          366#    376
TMRCNT  002766R          365*    375*    635#
TMRSET  001214R          155     365#
TRPDFD= 000022           190#
VA      002770R          266*    268     270*    271     636#
VECTOR  000010R          139#    217
WASADR  000104R          173#    506*
WDFR    000116R          180#    191*    201*
WDTO    000114R          179#    190*    200*
XFLAG   000005R          137#
XMTADR  002762R          272*    324     632#
XMTBUF  002426R          256     341     499     614#
XMTINT  001340R          413#    557     562     567     572     577     582     587     592
XMTQPI  002406R          246*    413*    414*    415     418*    605#
XMTQPO  002410R          248*    423     424*    425     427*    606#
XMTQUE  002646R          246     248     415     418     425     427     625#
XMTSRV  001400R          421     423#
.     = 003000R          599#    602#    614#    615#    616#    617#    618#    619#    620#    621#    622#    625#    626#
                         627#

. ABS.  000000      000
        003000      001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0
```

```
XDQAIO,XDQAIO/SOL/CRF:SYM=DDXCOM,XDQAIO
RUN-TIME:  1  2  .3 SECONDS
RUN-TIME RATIO: 15/4=3.2
CORE USED:  7K  (13 PAGES)
```