IDENTIFICATION

PRODUCT CODE: AC-E941C-MC

PRODUCT NAME: CXDZAC0 DZ11 MODULE

PRODUCT DATE: FEB 1979

MAINTAINER: DEC/X11 SUPPORT GROUP

1. ABSTRACT:

DZA IS AN IOMOD THAT EXERCISES UP TO EIGHT (CONSECUTIVELY ADDRESSED)
DZ11 ASYNCHRONOUS INTERFACES. IT USES MAINTENANCE MODE TO TRANSMIT
AND RECEIVE A BINARY COUNT PATTERN OUTPUTTED AND RECEIVED
IN 64. CHARACTER BURSTS. THE MAJOR PORTION OF THE ERROR
CHECKING IS DEFERRED TO PRIORITY LEVEL 0. ALL DEVICES SELECTED FOR TEST
ARE ACTIVATED AND RUN CONCURRENTLY. ALL EIGHT LINES ARE RUN ON EACH SELECTED DEVICE.

2. REQUIREMENTS:

HARDWARE: AT LEAST ONE DZ11 INTERFACE; NO WRAPAROUND OR MAINTENANCE CABLE IS NEEDED

STORAGE:: DZA REQUIRES:
1. DECIMAL WORDS: 1349
2. OCTAL WORDS: 02505
3. OCTAL BYTES: 5212

3. PASS DEFINITION:

ONE PASS OF THE DZA MODULE CONSISTS OF TRANSMITTING AND RE-
CEIVING 8960. CHARACTERS FOR EACH LINE OF EACH DZ11 SELECTED

4. EXECUTION TIME:

EXECUTION TIME IS PROPORTIONAL TO THE BAUD RATE BUT SHOULD TAKE AN AVERAGE OF ONE
MINUTE TO COMPLETE ONE PASS WHEN RUNNING ALONE ON A PDP11/40 AT 9600. BAUD.

5. CONFIGURATION PARAMETERS:

DEFAULT PARAMETERS:

DVA: 1, VCT: 1, BR1: 5, BR2: 5, DVC: 1, SR1: 0

REQUIRED PARAMETERS:

AT CONFIGURATION TIME THE USER MUST SPECIFY:

DVA:	ADDRESS OF FIRST DZ11 CSR REG.
VCT:	VECTOR ADDRESS OF FIRST DZ11
DVC:	NO OF DZ11'S IF GREATER THAN 1

6. DEVICE OPTION SETUP:

NONE REQUIRED

7. MODULE OPERATION:

START: DETERMINE IF ANY DEVICES ARE SELECTED. DO NOT RUN THE MODULE IF NO DEVICES ARE SELECTED. IF THERE ARE SELECTED DEVICES, INITIALIZE THE BINARY COUNT PATTERN AT 0. CONTINUE PROCESSING.

RESTRT: INITIALIZE THE ITERATION COUNTER TO 1120. DETERMINE IF ANY DZ11'S ARE SELECTED. LOAD THE INTERRUPT VECTORS TO POINT TO THE JSR LINKING TABLE.

SETUP2: INITIALIZE THE QUEUE POINTERS. CLEAR ALL THE BUFFERS AND QUEUES. CLEAR THE BUFFER ACCESS FLAG (LCKOUT) IN CASE IT WAS STILL SET BY A CONTROL C INTERRUPT OF THE PROGRAM.

ACTVATE: THIS SEGMENT INITIALIZES EACH DZ11 SELECTED. EIGHT BITS PER CHARACTER IS SELECTED. IF A BAUD RATE IS SELECTED, IT IS CALCULATED AND ASSIGNED. OTHERWISE THE DEFAULT RATE OF 9600 BAUD IS ASSUMED.

INITIAL: THE DATA PATTERN IS LOADED INTO THE TRANSMITTER BUFFER. IT IS A BINARY COUNT PATTERN WHICH, ON SUCCESSIVE ITERATIONS, BEGINS 0,10,20,....,177760,177770. THE NUMBER OF CHARACTERS TO BE TRANSMITTED IS CALCULATED. TRANSMITTER INTERRUPTS ARE ENABLED. ALL SELECTED TRANSMITTERS FOR EACH SELECTED DZ11 ARE ENABLED.

TMRSET: TMRCNT IS USED AS A MULTIPLYING FACTOR TO DETERMINE THE WAITING LENGTH FOR THE WATCHDOG TIMER. IT IS PRESENTLY SET AT 5 TO ALLOW SEVENTY-FIVE SECONDS TO ELAPSE BEFORE TAKING FURTHER ACTION.

TIMER: THIS IS THE WATCHDOG TIMER LOOP. IT IS CONTROLLED BY R4 AND TMRCNT. IF ALL DZ11'S SELECTED GENERATED BOTH TRANSMIT AND RECEIVE INTERRUPTS, THE APPROPRIATE BIT IN DONFLG FOR THAT DZ11 WILL BE CLEARED. IF THIS DOES NOT OCCUR IN THE GIVEN TIME, THE DEVICE NUMBER OF THE OFFENDING DZ11 WILL BE CALCULATED, AND THIS WILL BE REPORTED IN A MODULE MESSAGE. THE OFFENDING DEVICE IS DROPPED FROM THE EXERCISE. IF NO MORE DZ11'S ARE SELECTED, THE MODULE ITSELF IS DROPPED FROM THE RUN. IF MORE REMAIN TO BE EXERCISED, HOWEVER, CONTROL IS TRANSFERRED TO "FINISH."

FINISH: CONTROL COMES HERE IF ALL SELECTED DZ11'S WERE
SUCCESSFULLY EXERCISED OR IF MORE DZ11'S REMAIN
AFTER ONE WAS HUNG. THE ITERATION COUNT IS DECREASED.
IF THE COUNT DOES NOT REACH ZERO, CONTROL IS PASSED
TO SETUP2 AND THE MODULE IS RUN AGAIN. WHEN THE
COUNT REACHES ZERO, AN END OF PASS IS SIGNALLED.

XMTINT: THIS SEGMENT SAVES A POINTER TO THE CSR ADDRESS OF
THE INTERRUPTING DZ11 IN A FIRST-IN, FIRST-OUT,
WRAPAROUND BUFFER. THE TRANSMITTER QUEUE. THE ENTRY
POINTER IS THEN UPDATED TO POINT TO THE NEXT ENTRY
IN THE QUEUE.

XMTSRV: THIS BLOCK FETCHES A POINTER TO A CSR ADDRESS
FROM THE TRANSMITTER QUEUE, AND THE QUEUE IS UPDATED
TO THE NEXT ENTRY. THE CSR IS TESTED TO DETERMINE
WHAT KIND OF INTERRUPT OCCURRED. FALSE INTERRUPT IS
REPORTED. IF EVERYTHING IS CORRECT, THE ADDRESS OF
THE BYTE ASSOCIATED WITH THIS LINE (IN THE TRANSMITTER
BUFFER) IS CALCULATED. A CHARACTER IS TRANSMITTED
ON THIS LINE. TRANSMITTER INTERRUPTS ARE REENABLED
FOR THIS DEVICE, IF MORE CHARACTERS ARE TO BE TRANS-
MITTED. IF NO MORE ARE TO BE SENT ON THIS LINE, A
BIT POINTER IS BUILT AND THE TRANSMITTER IS DISABLED.
IF ALL LINES ARE DISABLED,THE RECEIVER FOR
THIS DZ11 IS ENABLED.

RCVINT: THIS SEGMENT SAVES A POINTER TO THE CSR ADDRESS
OF THE INTERRUPTING DZ11 IN THE RECEIVER QUEUE.
IT DISABLES FURTHER INTERRUPTS FOR THIS DEVICE.
IT UPDATES THE QUEUE ENTRY AND RESTORES THE VALUE
OF R5 WHICH WAS SAVED BY THE JSR INSTRUCTION FROM
THE LINKAGE TABLE.

RCVSRV: THE FIRST TASK IS TO PREVENT VOLATILE REGISTER
INFORMATION FROM BEING DESTROYED. THIS IS DONE
BY TESTING A SEMAPHORE, "LCKOUT." IF IT IS SET,
CONTROL IS RETURNED TO THE MONITOR TO WAIT FOR
A WHILE. IF IT IS CLEAR, ACCESS IS PERMITTED. THE
FLAG IS SET TO DENY OTHER ACCESSES TO THIS DEFERRED
ROUTINE. A CSR ADDRESS IS OBTAINED FROM THE QUEUE,
AND THE QUEUE ENTRY IS UPDATED. THE RECEIVER INTERRUPT
AND INTERRUPT ENABLE ARE CLEARED. THE REGISTERS ARE SET
UP TO RETRIEVE AS QUICKLY AS POSSIBLE THE DATA FROM
THE DZ11 SILO. EACH FETCH IS CHECKED TO SEE IF THE
INFORMATION IS VALID. IF IT IS NOT, THE REGISTERS
ARE SAVED AND A BREAK LOOP IS USED TO ALLOW MORE
TIME FOR VALID INFORMATION TO BECOME AVAILABLE.
IF AFTER THE ALLOTTED TIME ALL THE CHARACTERS ARE
STILL NOT RECEIVED, AN ERROR MESSAGE IS REPORTED.
IN THE MESSAGE, TWO NUMBERS ARE GIVEN. THE FIRST
IS THE NUMBER OF CHARACTERS THAT WERE TRANSMITTED
(IN OCTAL). THE NEXT IS THE NUMBER OF CHARACTERS
THAT WERE NOT RECEIVED (ALSO IN OCTAL).

CKDATA: THIS SEGMENT INITIALIZES THE LINE CHECK BUFFER

(LNCKBF) TO THE FIRST DATUM THAT WAS TRANSMITTED. THE DEVICE NUMBER IS SAVED FOR LATER USAGE. THE RECEIVED INFORMATION IS CHECKED FOR VALIDITY AND TRANSMISSION ERRORS. ERRORS ARE HANDLED BY THE 'STATERR' (STATUS ERROR) AND 'DERROR' (DATA ERROR) ROUTINES.

RCVDONE: THIS BLOCK CLEARS THE ACCESS SEMAPHORE TO ALLOW OTHER DEVICES TO USE THE LINE CHECK BUFFER. IT ALSO DISABLES THE DEVICE WITH A DEVICE CLEAR. IT THEN BUILDS A ONE BIT MASK USING R0 AND THE CARRY BIT TO DELETE THE APPROPRIATE BIT IN THE WATCHDOG TIMER FLAG (DONFLG). WHEN THIS IS DONE, PROCESSING CONTROL IS RETURNED TO THE MONITOR.

SUBROUTINES

VCTLOAD: THIS ROUTINE IS CALLED IN 'SETUP1'. IT IS USED TO LOAD THE ADDRESS OF THE LINKING INSTRUCTION FOR INTERRUPT SERVICING INTO THE CORRESPONDING VECTOR SPACE. IT ALSO LOADS THE PRIORITY LEVEL AND THE DEVICE ADDRESS. THE LATTER IS LOADED INTO THE APPROPRIATE JSR TABLE ENTRY.

SAVREG: THIS ROUTINE SAVES THE FIVE VOLATILE INFORMATION REGISTERS IN A FIRST-IN, FIRST-OUT WRAPAROUND BUFFER, THE ERROR QUEUE.

GETREG: THIS ROUTINE RETRIEVES THOSE SAME REGISTERS.

GETLINE: THIS ROUTINE CALCULATES HOW MANY CHARACTERS WILL BE TRANSMITTED FOR EACH DEVICE DURING AN ITERATION OF THE PROGRAM. EIGHT CHARACTERS ARE SENT ON EACH SELECTED LINE IN ONE ITERATION. THE TOTAL COUNT IS STORED IN 'XMTCNT'.

BAUDRTE: THIS ROUTINE CALCULATES THE BAUD RATE, ASSIGNS IT, AND SELECTS 8 BITS/CHARACTER COMMUNICATION MODE. IF SR1=0, THE DEFAULT RATE OF 9600. BAUD IS ASSIGNED. THE BAUD RATE SELECTED IS DETERMINED BY THE LEAST SIGNIFICANT (RIGHTMOST) SET BIT IN SR1.

STATERR: THIS ROUTINE DETERMINES WHETHER AN ERROR INDICATED IN THE RECEIVED CHARACTER INFORMATION WAS AN OVERRUN ERROR, A FRAMING ERROR, OR A PARITY ERROR. THE DEVICE NUMBER OF THE ERRING DEVICE IS REPORTED AS STATC. CSRA WILL BE CLEAR.

DERROR: THIS ROUTINE REPORTS A DATA ERROR.

8. OPERATOR OPTIONS

MODULE LOCATION DVID1 (APC=14) MAY BE MODIFIED (MOD. CMMD) TO EXERCISE
ANY COMBINATION OF EIGHT DZ11'S.

MODULE LOCATION SR1 (APC=16) MAY BE MODIFIED TO SELECT A DIFFERENT BAUD RATE.
THE FOLLOWING TABLE SHOULD BE USED:

FOR THE BAUD RATE	SR1=	LOC 336=
7200	1	2060
4800	2	2000
3600	4	1730
2400	10	1460
2200	20	1350
1800	40	1350
1200	100	1200
600	200	630
300	400	330
150	1000	150
134.5	2000	144
110	4000	120
75	10000	70
50	20000	45

(LOCATION 336 IS THE LOCATION OF THE ITERATION CONSTANT. USING THESE VALUES
WILL YIELD AN END OF PASS CLOSE TO ONE MINUTE FO EACH BAUD RATE.)

THE DEFAULT RATE IS 9600 BAUD(SR1=0).

MODULE LOCATION SLCTLIN (APC=310) MAY BE MODIFIED TO RUN ANY
COMBINATION OF EIGHT LINES. THE COMBINATION IS THEN RUN ON
ALL SELECTED DEVICES. THE DEFAULT SELECTION IS ALL EIGHT LINES.
NOTE: SLCTLIN FALLS ON A BYTE BOUNDARY!! BE SURE
TO RESTORE THE BITS SET IN THE OTHER BYTE !!!

MODULE LOCATION RESTRT +2 (APC=336) MAY BE MODIFIED TO VARY THE PERIOD BETWEEN END OF PASS REPORTS.

MODULE LOCATION TMRSET +2 (APC=752) MAY BE MODIFIED TO VARY
THE PERIOD OF THE WATCHDOG TIMER. IT IS PRESENTLY SET TO EXPIRE
AFTER SEVENTY-FIVE SECONDS WHEN DZA IS RUNNING ALONE.

9. NON-STANDARD PRINTOUTS:

WHEN A STATUS ERROR IS DETECTED, DZA USES THE ERRORN
CALL TO REPORT IT. THE FIRST NUMBER GIVEN IS THE NUMBER
OF THE DEVICE (0 TO 7). THE SECOND IS THE CONTENTS OF
THE READ BUFFER (DZRDBUF= CSR + 2,E.G.,160042).

WHEN ALL CHARACTERS ARE NOT RECEIVED, AN ERRORN CALL IS RE-
PORTED. THE FIRST NUMBER IS THE NUMBER OF CHARACTERS EXPECTED.
THE SECOND IS THE NUMBER OF CHARACTERS THAT WERE NOT RECEIVED.

ALL OTHER PRINTOUT IS STANDARD.

10. MNEMONICS

THE FOLLOWING INFORMATION SHOULD BE USEFUL IN UNDERSTANDING
NAMES GIVEN TO VARIABLES IN THIS PROGRAM.

XMT REFERS TO THE TRANSMITTER
RCV REFERS TO THE RECEIVER
ERR REFERS TO ANYTHING TO DO WITH ERROR HANDLING
FLG REFERS TO A SOFTWARE FLAG, USUALLY A BIT FLAG
QUE REFERS TO A FIRST IN, FIRST OUT BUFFER
TMR REFERS TO SOFTWARE TIMING FUNCTIONS
CNT REFERS TO A WORD USED AS A COUNTER
QP REFERS TO A POINTER ASSOCIATED WITH A QUEUE BUFFER
I IS AN INSERTION POINTER, O IS AN OUTPUT POINTER
LN REFERS TO SOMETHING INVOLVING A GIVEN LINE
XM IS ANOTHER REFERENCE TO TRANSMITTER
CT GENERALLY REFERS TO A COUNT

E.G:XMTQPO=TRANSMITTER QUEUE POINTER OUT

OTHERS ARE BASICALLY SELF EXPLANATORY.

```
                        IOMOD   <DZAC >1,1,5,5,,300,77
                        MODULE  140000,DZAC ,1,1,5,5,,300,77
                        .TITLE  DZAC DEC/X11 SYSTEM EXERCISER MODULE
                ;       DDXCOM  VERSION  6      23-MAY-78
                                .LIST   BIN
                        ;*****************************************************************************
000000'                 BEGIN:
000000' 055104  041501      040 MODNAM: .ASCII  /DZAC / ;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' 000352'                 INIT:   START               ;MODULE START ADDR.
000032' 000224'                 SPOINT: MODSP               ;MODULE STACK POINTER.
000034' 000000                  PASCNT: 0                   ;PASS COUNTER.
000036' 000300                  ICONT:  300                 ;# OF ITERATIONS PER PASS=300
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                  ASB:    OPEN                ;EXPECTED DATA.
000110' 000000                  AWAS:   OPEN                ;ACTUAL DATA.
000112' 000446'                 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' 000077                  IDNUM:  77                  ;MODULE IDENTIFICATION NUMBER=77
        000040                          .REPT   SPSIZ       ;MODULE STACK STARTS HERE.
                                        .NLIST
```



```
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224'                         MODSP:
                                ;*****************************************************************************
  407           000002                  RBUF=2              ;DEFINITION OF MODE 6 OFFSET TO THE READ BUFFER
  408           000002                  LPR=2               ;MODE 6 OFFSET TO THE LINE PARAMETER REGISTER
  409           000004                  TCR=4               ;OFFSET TO THE TRANSMITTER CONTROL REGISTER
  410           000006                  MSR=6               ;OFFSET TO THE MODEM STATUS REGISTER
  411           000006                  TDR=6               ;OFFSET TO THE TRANSMITTER DATA REGISTER
  412           000010                  MAINT=BIT3          ;ENABLE MAINTENANCE MODE BIT
  413           000020                  DCLR=BIT4           ;DEVICE MASTER CLEAR
  414           000040                  MSENAB=BIT5         ;ENABLE THE MASTER SCANNER (DEVICE GO)
  415           000100                  RIE=BIT6            ;RECEIVER INTERRUPT ENABLE
  416           010000                  SILOEN=BIT12        ;SILO ALARM INTERRUPT ENABLE
  417           040000                  TIE=BIT14           ;TRANSMITTER INTERRUPT ENABLE
  418           010000                  RCVON=BIT12         ;TURN THE RECEIVER ON (RECEIVER GO)
  419           000030                  EIGHT=BIT4!BIT3     ;EIGHT BITS/CHARACTER SELECTION
```

```
                        ;THESE ARE THE PROGRAM PARAMETERS

000224' 000000          LINPAR: OPEN                    ;SCRATCHWORD USED TO LOAD LINE PARAMETER REGISTERS
000226' 000000          TMRCNT: OPEN                    ;COUNTER FOR WATCHDOG TIMER
000230' 000000          RCVTMR: OPEN                    ;RECEIVER BREAK LOOP TIMER
000232' 000000          DVCNMBR:OPEN                    ;NUMBER OF DEVICE BEING PROCESSED
000234' 000000          XMTCNT: OPEN                    ;COUNTER OF TOTAL NUMBER OF TRANSMISSIONS
000236' 000100          LNXMCT: .BLKB   64.             ;TRANSMISSION COUNTERS FOR EACH DEVICE
000336' 000000          BGNDATA:OPEN                    ;TWO COPIES OF THE FIRST CHARACTER IN TRANSMIT PATTERN
000340' 000000          RCVWORD:OPEN                    ;USED TO REPORT RECEIVER ERROR
000342' 000000          MISS:   OPEN                    ;USED TO REPORT NUMBER OF CHARACTERS MISSING
000344' 000             DATA:   .BYTE   OPEN            ;CHARACTER BUILDING BYTE
000345' 000             SELECT: .BYTE   OPEN            ;ACTIVE DEVICE SELECTION PARAMETER
000346' 377             SLCTLIN:.BYTE   377             ;ACTIVE LINE SELECTION PARAMETER
000347' 000             DONFLG: .BYTE   OPEN            ;WATCHDOG FLAG FOR BUSY DEVICES
000350' 000             RCVDATA:.BYTE   OPEN            ;BUFFER FOR CHARACTER CHECKING
000351' 000             LCKOUT: .BYTE   OPEN            ;BUFFER ACCESS FLAG
                                .EVEN
```



```
                        ;START ROUTINE. DETERMINE IF ANY DZ'S ARE SELECTED
                        ;IF SO, BEGIN MODULE PROCESSING... IF NOT, DROP THE MODULE FROM THE RUN

000352' 012767 000010 177534 START:  MOV     #8.,WDTO           ;8 WORDS TO MEM/ITERATION
000360' 012767 000010 177530         MOV     #8.,WDFR           ;8 WORDS FROM MEM/ITERATION
000366' 016701 177422                MOV     DVID1,R1           ;DVC TO R1
000372' 006201               1$:     ASR     R1                 ;SHIFT IN A DEVICE
000374' 103402                       BCS     2$                 ;BR IF A DVC SELECTED HERE
000376' 001413                       BEQ     3$                 ;BR IF NONE LEFT
000400' 000774                       BR      1$                 ;GO BACK FOR MORE
000402' 062767 000010 177504 2$:     ADD     #8.,WDTO           ;ADD 8 MORE
000410' 062767 000010 177500         ADD     #8.,WDFR           ;DITTO
000416' 062767 000010 177474         ADD     #8.,INTR           ;8 MORE INTERRUPTS
000424' 000762                       BR      1$                 ;GO SEE IF MORE DEVS
000426' 105067 177712        3$:     CLRB    DATA               ;INITIALIZE THE DATA PATTERN WORD
000432' 116767 177356 177705         MOVB    DVID1,SELECT       ;COPY THE DEVICE SELECTION PARAMETER. ARE
                                                                ;ANY DEVICES SELECTED?
000440' 001002                       BNE     RESTRT             ;IF SO, BEGIN PROCESSING. IF NOT, DROP THE MODULE
000442'                      DROP:
000442' 104410 000000'               ENDS,BEGIN                 ;DROP THE MODULE

                        ;INITIALIZE THE NUMBER OF PASSES TO BE MADE. SET UP THE DZ11 INTERRUPT
                        ;VECTORS TO INTERRUPT IN THE SUBROUTINE LINKING TABLE. CALL SUBROUTINE
                        ;VCTLOAD FOR THIS PURPOSE.

000446'                 RESTRT:
000446' 116701 177673   SETUP1: MOVB    SELECT,R1          ;COPY THE DEVICE SELECTION PARAMETER INTO R1
000452' 001773                   BEQ     DROP               ;IF NO DEVICES SELECTED (ALL FLAGS CLEAR) DROP THE
                                                            ;MODULE
000454' 016700 177330            MOV     VECTOR,R0          ;LOAD THE VECTOR ADDRESS IN R0
000460' 016702 177322            MOV     ADDR,R2            ;LOAD R2 WITH ADDRESS OF FIRST DZ11
000464' 012703 002476'           MOV     #LNKTAB,R3         ;POINT R3 TO THE BEGINNING OF JSR LINKING TABLE
000470' 000241                   CLC                        ;MAKE SURE CARRY BIT IS CLEAR BEFORE ROTATION
000472' 106001               1$: RORB    R1                 ;ISOLATE A SELECTION FLAG IN THE CARRY BIT
000474' 103410                   BCS     3$                 ;IF THE FLAG IS SET, GO SET UP THE VECTORS
000476' 001416                   BEQ     SETUP2             ;IF NO FLAGS ARE LEFT, GO SET UP THE BUFFERS
000500' 062700 000010            ADD     #10,R0             ;IF MORE FLAGS ARE SET, ADJUST POINTERS. THE
                                                            ;VECTOR POINTER...
000504' 062703 000020            ADD     #20,R3             ;THE LINKAGE TABLE POINTER...
000510' 062702 000010       2$:  ADD     #10,R2             ;AND THE ADDRESS POINTER
000514' 000766                   BR      1$                 ;GO SET UP THE NEXT DZ11 ADDRESSES
000516' 004567 001406       3$:  JSR     R5,VCTLOAD         ;CALL THE VECTOR SETUP ROUTINE FOR RECEIVER
000522' 000013'                  BR2                        ;VECTOR, PASSING THE RECEIVER BR LEVEL AS THE
                                                            ;ARGUMENT
000524' 004567 001400            JSR     R5,VCTLOAD         ;SET UP THE TRANSMITTER VECTOR PASSING THE
000530' 000012'                  BR1                        ;TRANSMITTER BR LEVEL AS THE ARGUMENT
000532' 000766                   BR      2$                 ;GO SETUP THE NEXT DZ11

                        ;THIS BLOCK RESETS ALL THE QUEUE POINTERS, CLEARS THE TRANSMITTER TEXT BUFFER, THE
                        ;RECEIVER BUFFERS, AND THE QUEUES. THIS IS THE BEGINNING OF THE ITERATIVE PART OF THE
                        ;PROGRAM.

000534' 012767 002676' 004366 SETUP2: MOV     #XMTQUE,XMTQPI ;POINT THE TRANSMIT QUEUE ENTRY (IN) POINTER
000542' 012767 002676' 004362        MOV     #XMTQUE,XMTQPO ;TO THE BEGINNING OF THE QUEUE
                                                            ;POINT THE TRANSMIT QUEUE RETRIEVAL (OUT)
                                                            ;POINTER TO THE BEGINNING OF THE QUEUE
```

```
000550' 012767  002716' 004356          MOV     #RCVQUE,RCVQPI  ;SET UP THE RECEIVER QUEUE POINTERS
000556' 012767  002716' 004352          MOV     #RCVQUE,RCVQPO  ;
000564' 012767  002736' 004346          MOV     #ERRQUE,ERRQPI  ;SETUP THE ERROR QUEUE POINTERS
000572' 012767  002736' 004342          MOV     #ERRQUE,ERRQPO  ;
000600' 012703  002676'                 MOV     #XMTQUE,R3      ;POINT R3 TO THE BEGINNING OF THE QUEUE AREA
000604' 012704  001115                  MOV     #589.,R4        ;USING R4 AS A COUNTER CLEAR -
000610' 005023                  1$:     CLR     (R3)+           ;TRANSMITTER, RECEIVERS, AND ERROR QUEUES....
000612' 005304                          DEC     R4              ;TRANSMITTER TEXT BUFFER...
000614' 001375                          BNE     1$              ;RECEIVER SILO BUFFERS
000616' 105067  177527                  CLRB    LCKOUT          ;CLEAR THE BUFFER ACCESS FLAG
000622' 012703  000236'                 MOV     #LNXNCT,R3      ;POINT TO THE CHRACTER COUNTER FOR EACH LINE
000626' 012704  000100                  MOV     #64.,R4         ;SET UP 64. BYTE POINTERS
000632' 112723  000010          2$:     MOVB    #8.,(R3)+       ;SET UP THIS BYTE AND POINT TO THE NEXT ONE
000636' 005304                          DEC     R4              ;REDUCE THE COUNT. ARE 64. BYTES SET UP?
000640' 001374                          BNE     2$              ;IF NO, GO SET UP THE NEXT ONE

                                ;THIS BLOCK SETS THE WATCHDOG TIMER FLAG, INITIALIZES THE DZ11 CONDITIONS, SETS THE CHAR
                                ;AND BAUD RATE FOR THE ACTIVE DZ11S

000642' 116700  177477          ACTVATE:MOVB    SELECT,R0       ;COPY THE DEVICE SELECTION PARAMETER
000646' 110067  177475                  MOVB    R0,DONFLG       ;SET THE WATCHDOG TIMER DEVICE COMPLETION FLAG
000652' 016701  177130                  MOV     ADDR,R1         ;LOAD THE ADDRESS OF THE FIRST DZ11
000656' 106000                  1$:     RORB    R0              ;ISOLATE A SELECTION FLAG IN THE CARRY BIT
000660' 103404                          BCS     3$              ;IF SELECTED, GO SET UP THE DEVICE REGISTERS
000662' 001426                          BEQ     INITIAL         ;IF NO MORE SELECTED, GO SET UP THE DATA
000664' 062701  000010          2$:     ADD     #10,R1          ;POINT R1 TO THE NEXT DZ11
000670' 000772                          BR      1$              ;PROCESS NEXT DZ11
000672' 012711  000020          3$:     MOV     #DCLR,(R1)      ;MASTER CLEAR THIS DZ11
000676' 012702  000010                  MOV     #10,R2          ;USE R2 TO COUNT 8. LINES BEING SET UP
000702' 012767  010030  177314          MOV     #RCVON!EIGHT,LINPAR ;SET UP NO PARITY, EIGHT BITS/CHAR.
000710' 004767  001366                  JSR     PC,BAUDRTE      ;GO CALCULATE THE BAUD RATE TO BE USED
000714' 052711  000010                  BIS     #MAINT,(R1)     ;ENABLE MAINTENANCE MODE OPERATION
000720' 016761  177300  000002  4$:     MOV     LINPAR,LPR(R1)  ;SET UP THIS LINE'S PARAMETERS
000726' 005302                          DEC     R2              ;REDUCE THE COUNT. ARE ALL 8. LINES SET?
000730' 001755                          BEQ     2$              ;IF YES, GO PROCESS NEXT DZ11
000732' 005267  177266                  INC     LINPAR          ;IF NO, INCREMENT THE LINE SELECTION PARAMETER
000736' 000770                          BR      4$              ;GO SET UP THE NEXT LINE

                                ;THIS BLOCK SETS UP THE TRANSMITTER TEXT WITH THE TEST DATA . IT THEN STARTS EACH
                                ;RECEIVER AND EACH TRANSMITTER SELECTED

000740' 012701  003020'         INITIAL:MOV     #XMTBUF,R1      ;POINT R1 TO THE START OF THE BUFFER AREA
000744' 116700  177374                  MOVB    DATA,R0         ;BEGIN SETTING UP THE CHARACTER PATTERN
000750' 110067  177362                  MOVB    R0,BGNDATA      ;COPY THE FIRST DATUM BEING TRANSMITTED IN THIS
000754' 110067  177357                  MOVB    R0,BGNDATA+1    ;GROUP .BGNDATA WILL BE USED TO SET
                                                                ;UP LNCKBF(THE LINE CHECK BUFFER ) IN THE CKDATA ROUTINE
000760' 012702  000100                  MOV     #64.,R2         ;USE R2 TO COUNT THE LOOP ITERATIONS
000764' 110021                  1$:     MOVB    R0,(R1)+        ;PUT A CHARACTER IN THE BUFFER
000766' 005302                          DEC     R2              ;REDUCE COUNT. HAVE ALL CHARACTERS BEEN MADE?
000770' 001375                          BNE     1$              ;IF NO, GO LOAD THE NEXT CHARACTER
000772' 062700  000010                  ADD     #8.,R0          ;POINT TO THE BEGINNING OF THE NEXT PATTERN
000776' 110067  177342                  MOVB    R0,DATA         ;SAVE THAT PATTERN BEGINNING
001002' 116700  177337                  MOVB    SELECT,R0       ;COPY THE DEVICE SELECTION PARAMETER
001006' 016701  176774                  MOV     ADDR,R1         ;LOAD THE ADDRESS OF THE FIRST DZ11
001012' 004767  001224                  JSR     PC,GETLINE      ;FIND OUT HOW MANY LINES ARE RUNNING
001016' 000241                          CLC                     ;MAKE SURE CARRY BIT IS CLEAR BEFORE ROTATION
```



```
001020' 106000          2$:     RORB    R0              ;ISOLATE A SELECTION FLAG IN THE CARRY BIT
001022' 103404                  BCS     4$              ;IF SELECTED, GO START THE DEVICE
001024' 001411                  BEQ     TMRSET          ;IF NO MORE SELECTED, GO START WATCHDOG TIMER
001026' 062701  000010  3$:     ADD     #10,R1          ;POINT R1 TO THE NEXT DZ11
001032' 000772                  BR      2$              ;GO PROCESS THE NEXT DZ11
001034' 052711  040040  4$:     BIS     #TIEIMSENAB,(R1);ENABLE TRANSMITTER INTERRUPTS
001040' 116761  177302  000004          MOVB    SLCTLIN,TCR(R1) ;ENABLE THE TRANSMITTERS FOR ALL SELECTED LINES
001046' 000767                  BR      3$              ;PROCESS THE NEXT DZ11
```

```
                                        ;THIS IS THE WATCHDOG TIMER. WHEN A DEVICE HAS SUCCESSFULLY TRANSMITTED AND
                                        ;RECEIVED ALL DATA. IT CLEARS ITS CORRESPONDING BIT IN DONFLG. IF THIS
                                        ;DOES NOT OCCUR, A MODULE MESSAGE IS REPORTED. IF MORE DEVICES ARE TO BE
                                        ;PROCESSED, THIS ITERATION IS CONSIDERED COMPLETE. IF NO DEVICES REMAIN TO BE
                                        ;PROCESSED, THE MODULE IS DROPPED.

001050' 012767  000005  177150  TMRSET: MOV     #5,TMRCNT       ;SET THE TIMER COUNT FOR THE BREAK LOOP
001056' 005004                  TIMER:  CLR     R4              ;USING R4, RETURN TO MONITOR 65535 TIMES
001060'                         1$:
001060' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001064' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001070' 105767  177253                  TSTB    DONFLG          ;IF DONFLG IS CLEAR, EACH SELECTED DEVICE WAS
                                                                ;SUCCESSFUL
001074' 001437                  BEQ     FINISH          ;IF SO, PERFORM ENDPASS PROCESSING
001076' 005304                  DEC     R4              ;IF NOT, REDUCE COUNT AND BREAK AGAIN
001100' 001367                  BNE     1$              ;BREAK IF COUNT NOT EXCEEDED
001102' 005367  177120          DEC     TMRCNT          ;REDUCE THE OVERALL TIME.IS IT EXCEEDED?
001106' 001363                  BNE     TIMER           ;IF NO,START ANOTHER BREAK LOOP

001110' 116703  177233          MOVB    DONFLG,R3       ;IF TIMEOUT OCCURRED, SAVE THE REMAINING FLAG
                                                        ;IN R3.
001114' 140367  177225          BICB    R3,SELECT       ;CLEAR THE SELECTION FLAG FOR THIS DEVICE
001120' 006003                  2$:     ROR     R3              ;DETERMINE WHICH DEVICE WAS READ FOR REPORTING
                                                                ;PURPOSES
001122' 103402                  BCS     3$              ;IF THIS IS THE DEVICE, R4 CONTAINS THE CORRECT
                                                                ;LINE NUMBER... GO REPORT IT
001124' 005204                  INC     R4              ;IF NOT, INCREMENT R4, WHICH WAS INITIALLY 0
                                                                ;FROM THE PREVIOUS LOOP
001126' 000774                  BR      2$              ;GO SEE IF THIS IS THE DEVICE
001130' 010467  004010          3$:     MOV     R4,NUMBA1       ;SAVE IT
                                        ;*****************************************************
                                                                ;CONVERT NUMBA1 TO ASCII AND
                                                                ;STORE AT M2
001134' 104420  000000' 005144'         OTOA$,BEGIN,NUMBA1,M2
001142' 005176'
                                        ;*****************************************************************
001144' 004767  000776                  JSR     PC,SAVREG       ;SAVE THE REGISTERS BEFORE LEAVING THE MODULE
001150' 104403  000000' 005162'         MSGN$,BEGIN,HUNG        ;ASCII MESSAGE CALL WITH COMMON HEADER
001156' 004767  001022                  JSR     PC,GETREG       ;RESTORE THE PREVIOUS REGISTER VALUES
001162' 105767  177157                  TSTB    SELECT          ;ARE THERE ANY DEVICES STILL ACTIVE?
001166' 001002                  BNE     FINISH          ;IF YES, REDUCE COUNT AND CONTINUE
                                                                ;IF NO, DROP THE MODULE
001170' 104410  000000'                 END$,BEGIN              ;DROP THE MODULE

                                        ;THIS BLOCK REDUCES THE ITERATION COUNT AND GOES TO ACTVATE IF THE COUNT IS NOT EXCEEDED

001174'                         FINISH:
001174' 104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                ;MONITOR SHALL TEST END OF PASS
001200' 000167  177436                  JMP     ACTVATE         ;IF NO, START PROCESSING AGAIN

                                        ;TRANSMITTER INTERRUPT SERVICE ROUTINE
                                        ;THIS ROUTINE STORES THE ADDRESS OF THE CSR POINTER (OFFSET IN THE JSR LINKAGE
                                        ;TABLE) IN THE TRANSMITTER QUEUE
                                        ; THE ROUTINE THEN ENABLES INTERRUPTS FOR THE CORRESPONDING RECEIVER
                                        ;AND RETURNS CONTROL TO THE MONITOR.
```



```
001204' 010577  003720          XMTINT: MOV     R5,@XMTQPI      ;LOAD THE OFFSET TO THE CSR INTO TRANSMITTER QUEUE
001210' 062767  000002  003712          ADD     #2,XMTQPI       ;UPDATE THE QUEUE ENTRY POINTER
001216' 022767  002714' 003704          CMP     #XMTQUE+16,XMTQPI ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001224' 103003                  BHIS    1$              ;IF NO, CONTINUE PROCESSING
001226' 012767  002676' 003674          MOV     #XMTQUE,XMTQPI  ;IF YES, RESET THE POINTER TO QUEUE BEGINNING
001234' 012605                  1$:     MOV     (SP)+,R5        ;RESTORE THE PREVIOUS R5 VALUE
001236' 010046                  MOV     R0,-(SP)
001240' 010146                  MOV     R1,-(SP)
001242' 010246                  MOV     R2,-(SP)
001244' 010346                  MOV     R3,-(SP)
001246' 010446                  MOV     R4,-(SP)
001250' 017701  003656          XMTSRV: MOV     @XMTQPO,R1      ;FETCH THE OFFSET FROM THE QUEUE
001254' 062767  000002  003650          ADD     #2,XMTQPO       ;UPDATE THE QUEUE RETRIEVAL POINTER
001262' 022767  002714' 003642          CMP     #XMTQUE+16,XMTQPO ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001270' 103003                  BHIS    1$              ;IF NO, CONTINUE PROCESSING
001272' 012767  002676' 003632          MOV     #XMTQUE,XMTQPO  ;IF YES, RESET POINTER TO BEGINNING OF THE QUEUE
001300' 011100                  1$:     MOV     (R1),R0         ;LOAD CSR ADDRESS INTO R0
001302' 005710                  TST     (R0)            ;IS THIS A VALID INTERRUPT?
001304' 100430                  BMI     2$              ;IF YES, GO ENABLE RECEIVER INTERRUPT
001306' 010067  176566                  MOV     R0,CSRA         ;IF NO, SET UP THE CSR ADDRESS FOR ERROR REPORTING
001312' 011067  176564                  MOV     (R0),ACSR       ;SET UP THE CONTENTS
001316' 004767  000624                  JSR     PC,SAVREG       ;SAVE THE REGISTERS BEFORE ERROR MESSAGE
001322' 012604                  MOV     (SP)+,R4
001324' 012603                  MOV     (SP)+,R3
001326' 012602                  MOV     (SP)+,R2
001330' 012601                  MOV     (SP)+,R1
001332' 012600                  MOV     (SP)+,R0
                                        ;-------------------------------------------------------------
001334' 000004  000000' 001342'         PIRQ$,BEGIN,10$         ; QUEUE UP TO CONTINUE AT 10$ AND RTI
                                        ;-------------------------------------------------------------
001342' 012767  000011  176536  10$:    MOV     #11,ERRTYP      ;ILLEGAL INTERRUPT
                                        ;*************************************************************
001350' 104405  000000' 000000          ERDER$,BEGIN,NULL       ;FALSE INTERRUPT REQUEST FROM TRANSMITTER
                                        ;*************************************************************
001356' 004767  000622                  JSR     PC,GETREG       ;RESTORE THE REGISTERS
001362' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
001366' 116002  000001          2$:     MOVB    1(R0),R2        ;GET THE LINE NUMBER FROM THE HIGH BYTE OF CSR
001372' 042702  177770                  BIC     #177770,R2      ;ISOLATE THE LINE NUMBER IN R2
001376' 016103  000002                  MOV     2(R1),R3        ;GET THE NUMBER OF THIS DEVICE
001402' 006303                  ASL     R3              ;MULTIPLY THE DEVICE NUMBER BY EIGHT
001404' 006303                  ASL     R3              ;THIS INVOLVES MULTIPLYING IT BY TWO
001406' 006303                  ASL     R3              ; THREE TIMES IN A ROW
001410' 060203                  ADD     R2,R3           ;CALCULATE THIS LINE'S TABLE OFFSET
001412' 042710  040000                  BIC     #BIT14,(R0)     ;DISABLE INTERRUPTS BEFORE AN OVERRUN OCCURS
001416' 116360  003020' 000006          MOVB    XMTBUF(R3),TDR(R0) ;LOAD THE CHARACTER POINTED TO FOR THIS
                                                                ;LINE IN THE TRANSMITTER BUFFER
001424' 105263  003020'                 INCB    XMTBUF(R3)      ;CREATE THE NEXT CHARACTER
                                                                ;TO BE TRANSMITTED

001430' 105363  000236'                 DECB    LNXMCT(R3)      ;COUNT A CHARACTER TRANSMITTED FOR THIS LINE
                                                                ;HAVE 8. CHARACTERS BEEN TRANSMITTED?
001434' 001021                  BNE     5$              ;IF NO, GO GET OUT OF THIS ROUTINE
001436' 112763  000010  000236'         MOVB    #8.,LNXMCT(R3)  ;IF SO, RESET THE COUNT OF CHARACTERS
001444' 012703  000001                  MOV     #1,R3           ;SET A 1 BIT POINTER IN R3
001450' 005302                  3$:     DEC     R2              ;USE R2 AS A COUNT OF ROTATIONS TO MAKE
```

```
001452' 100402                  BMI     4$              ;IS THE POINTER NOW AT THE RIGHT LINE?
001454' 006303                  ASL     R3              ;IF NOT, POINT IT TO THE NEXT LINE
001456' 000774                  BR      3$              ;GO SEE IF THIS IS THE LINE
001460' 040360  000004  4$:     BIC     R3,TCR(R0)      ;IF SO, DISABLE TRANSMISSIONS ON THIS LINE
001464' 001005                  BNE     5$              ;IF MORE TRANSMITTERS ARE ACTIVE, GO EXIT
001466' 052710  000100          BIS     #TIE,(R0)       ;OTHERWISE, ENABLE THE RECEIVER
001472' 042710  040000          BIC     #TIE,(R0)       ;DISABLE TRANSMITTER INTERRUPTS
001476' 000402                  BR      6$              ;SKIP RESETTING THE INTERRUPT ENABLE
001500' 052710  040000  5$:     BIS     #TIE,(R0)       ;RESTART THE TRANSMITTER
001504' 012604          6$:     MOV     (SP)+,R4
001506' 012603                  MOV     (SP)+,R3
001510' 012602                  MOV     (SP)+,R2
001512' 012601                  MOV     (SP)+,R1
001514' 012600                  MOV     (SP)+,R0
001516' 000002                  RTI
```



```
                                ;RECEIVER INTERRUPT SERVICE ROUTINE
                        ;THIS ROUTINE STORES A POINTER TO THE CSR ADDRESS OF THE INTERRUPTING DEVICE IN THE
                        ;RECEIVER QUEUE. SERVICE IS DEFERRED TO PRIORITY LEVEL 0. AT DEFERRED SERVICE (RCVSRV),
                        ;THE OFFSET IS FETCHED FROM THE QUEUE. THE INTERRUPT AND INTERRUPT ENABLE BITS ARE
                        ;CLEARED AND THE SILO IS EMPTIED INTO THE SOFTWARE BUFFER. IF ALL CHARACTERS
                        ;WERE NOT RECEIVED, THIS IS REPORTED. CHECK THE BUFFER AND REPORT DATA ERRORS. IF
                        ;ANOTHER RECEIVER IS ALREADY USING THE LINE CHECK BUFFER (LNCKBF), SUBSEQUENT
                        ;RECEIVERS WILL NOT BE PERMITTED ACCESS UNTIL THE FIRST IS COMPLETE. IF THERE
                        ;ARE ANY CHARACTER ERRORS REPORT THEM. WHEN ALL DATA ARE CHECKED, IT RELEASES
                        ;THE LINE CHECK BUFFER AND CLEAR THE CORRESPONDING BIT IN DONFLG.

001520' 010577  003410          RCVINT: MOV     R5,@RCVQPI      ;STORE THE OFFSET IN THE RETRIEVAL QUEUE
001524' 042735  000100                  BIC     #RIE,@(R5)+     ;DISABLE THIS RECEIVER'S INTERRUPTS BEFORE
                                                                ;ITS OTHER LINES OVERRUN THE QUEUES
001530' 062767  000002  003376          ADD     #2,RCVQPI       ;UPDATE THE QUEUE ENTRY POINTER
001536' 022767  002734' 003370          CMP     #RCVQUE+16,RCVQPI ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001544' 103003                          BHIS    1$              ;IF NOT, CONTINUE PROCESSING
001546' 012767  002716' 003360          MOV     #RCVQUE,RCVQPI  ;RESET THE POINTER TO QUEUE BEGINNING
001554' 012605          1$:     MOV     (SP)+,R5        ;RESTORE THE PREVIOUS VALUE OF R5
                                ;-----------------------------------------------------------------
001556' 000004  000000' 001564'         PIRQ$,BEGIN,RCVSRV      ; QUEUE UP TO CONTINUE AT RCVSRV AND RTI
                                ;-----------------------------------------------------------------
001564' 105767  176561          RCVSRV: TSTB    LCKOUT          ;TEST THE BUFFER LOCK FLAG. IS ACCESS PERMITTED?
001570' 001405                  BEQ     11$             ;IF YES, GO SET THE LOCK AND CHECK THE DATA
001572' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001576' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001602' 000770                          BR      RCVSRV          ;TRY TO ACCESS AGAIN
001604' 105167  176541          11$:    COMB    LCKOUT          ;DENY OTHER ACCESSES TO THE BUFFER
001610' 017700  003322                  MOV     @RCVQPO,R0      ;FETCH THE OFFSET FROM THE QUEUE
001614' 062767  000002  003314          ADD     #2,RCVQPO       ;UPDATE THE QUEUE RETRIEVAL POINTER
001622' 022767  002734' 003306          CMP     #RCVQUE+16,RCVQPO ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001630' 103003                          BHIS    1$              ;IF NO, CONTINUE PROCESSING
001632' 012767  002716' 003276          MOV     #RCVQUE,RCVQPO  ;IF YES, RESET THE POINTER TO THE QUEUE BEGINNING
001640' 012001          1$:     MOV     (R0)+,R1        ;LOAD THE CSR ADDRESS INTO R1
001642' 042721  030300                  BIC     #30300,(R1)+    ;DISABLE RECEIVER INTERRUPT AND INTERRUPT ENABLE
                                                                ;AND POINT R1 TO THE NEXT RECEIVED CHAR. REGISTER
001646' 011002                          MOV     (R0),R2         ;LOAD THE SOFTWARE SILO BUFFER ADDRESS
001650' 012767  002000  176352          MOV     #2000,RCVTMR    ;SET THE RECEIVER BREAK LOOP TIMER
001656' 016703  176352                  MOV     XMTCNT,R3       ;SET THE COUNT FOR STORING THE DATA
001662' 011122          2$:     MOV     (R1),(R2)+      ;STORE A WORD IN THE SOFTWARE SILO
001664' 002003                          BGE     3$              ;IF THE DATA WASN'T VALID, GO ALLOW A LITTLE TIME FOR IT
001666' 005303                          DEC     R3              ;IF IT WAS VALID, REDUCE THE COUNT.
001670' 001374                          BNE     2$              ;IF NOT ZERO, GO STORE THE NEXT WORD
001672' 000434                          BR      CKDATA          ;IF ALL CHARACTERS RECEIVED, GO CHECK THEM
001674' 005742          3$:     TST     -(R2)           ;RESET R2 TO THE PROPER BUFFER LOCATION
001676' 004767  000244                  JSR     PC,SAVREG       ;SAVE THE REGISTER VALUES
001702' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001706' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001712' 004767  000266                  JSR     PC,GETREG       ;RESTORE THE REGISTER VALUES
001716' 005367  176306                  DEC     RCVTMR          ;HAS ENOUGH TIME BEEN PERMITTED FOR A CHARACTER?
001722' 100357                          BPL     2$              ;IF NO, TRY TO GET ONE.
001724' 014067  176150                  MOV     -(R0),CSRA      ;LOAD THE DEVICE ADDRESS
001730' 013067  176146                  MOV     @(R0)+,ACSR     ;LOAD THE CSR CONTENTS
001734' 010367  176402                  MOV     R3,MISS         ;REPORT THE NUMBER OF CHARACTERS MISSING
001740' 004767  000202                  JSR     PC,SAVREG       ;SAVE THE REGISTERS
```

```
001744' 012767  000001  176134          MOV     #1,ERRTYP       ;DATA ERROR
                                        ;*************************************
001752' 104405  000000' 005146'         ERRORS,BEGIN,TAB1       ;DID NOT RECEIVE ALL TRANSMITTED CHARACTERS
                                        ;*************************************
001760' 004767  000220                  JSR     PC,GETREG

                                ;THIS ROUTINE PERMITS ONE DEVICE TO HAVE ITS DATA CHECKED.
                                ;THE DATA IS CHECKED
                                ;FOR EACH LINE OF THE DEVICE. DATA ERRORS ARE REPORTED.

001764' 012703  000004          CKDATA: MOV     #4,R3           ;SET UP A COUNT FOR CLEARING THE BUFFER
001770' 012705  005120'                 MOV     #LNCKBF,R5      ;POINT TO THE BEGINNING OF THE BUFFER
001774' 016725  176336          2$:     MOV     RCVDATA,(R5)+   ;SET UP TWO LINE CHECK BYTES
002000' 005303                          DEC     R3              ;REDUCE THE COUNT. HAVE 8. BYTES BEEN SET UP ?
002002' 001374                          BNE     2$              ;IF NO, GO SET UP THE REMAINING
002004' 011002                          MOV     (R0),R2         ;LOAD THE SOFTWARE SILO ADDRESS IN R2
002006' 016067  000010  176216          MOV     10(R0),DVCNMBR  ;SAVE THE NUMBER OF THIS DEVICE
002014' 016703  176214                  MOV     XMTCNT,R3       ;LOAD COUNT TO COMPARE CHARACTERS
002020' 012200                  3$:     MOV     (R2)+,R0        ;FETCH A WORD FROM THE SILO INTO R0
002022' 002024                          BGE     RCVDONE         ;IF IT'S INVALID, GO CLEAN UP BUFFERS
002024' 032700  070000                  BIT     #70000,R0       ;ARE THERE ANY TRANSMISSION ERRORS?
002030' 001402                          BEQ     4$              ;IF NO, GO DETERMINE THE LINE NUMBER
002032' 004767  000322                  JSR     PC,STATERR      ;IF YES, GO DETERMINE THE ERROR TYPE
002036' 110067  176306          4$:     MOVB    R0,RCVDATA      ;SAVE THE RECEIVED CHARACTER
002042' 000300                          SWAB    R0              ;PUT THE LINE NUMBER IN RO'S LOWER BYTE
002044' 042700  177770                  BIC     #177770,R0      ;ISOLATE THE LINE NUMBER IN R0
002050' 126067  005120' 176272          CMPB    LNCKBF(R0),RCVDATA ;IS THE DATA CORRECT?
002056' 001402                          BEQ     5$              ;IF YES, GO CHECK THE NEXT CHARACTER
002060' 004767  000334                  JSR     PC,DERROR       ;IF NO, GO REPORT A DATA ERROR
002064' 105260  005120'         5$:     INCB    LNCKBF(R0)      ;SET UP THE NEXT CHARACTER FOR THIS LINE
002070' 005303                          DEC     R3              ;REDUCE THE COUNT. ARE ALL CHARACTERS CHECKED?
002072' 001352                          BNE     3$              ;IF NO, GO CHECK THE NEXT CHARACTER.

                                ;THIS ROUTINE UNLOCKS THE LINE CHECK BUFFER TO PERMIT ACCESS BY OTHER DEVICES.
                                ;IT THEN COMPUTES THE LINE NUMBER AND CLEARS THE APPROPRIATE FLAG FROM THE WATCHDOG
                                ;TIMER BYTE.

002074' 105067  176251          RCVDONE:CLRB    LCKOUT          ;RESET THE BUFFER ACCESS SEMAPHORE
002100' 012741  000020                  MOV     #DCLR,-(R1)     ;CLEAR THIS DEVICE AND DISABLE IT
002104' 005000                          CLR     R0              ;THE DELETION FLAG WILL PROPAGATE IN R0
002106' 000261                          SEC                     ;USE THE CARRY BIT TO BUILD THE DELETION FLAG
002110' 006100                  1$:     ROL     R0              ;MOVE THE FLAG TO THE NEXT DEVICE
002112' 005367  176114                  DEC     DVCNMBR         ;WHEN THIS NUMBER IS NEGATIVE, THE CORRECT
002116' 002374                          BGE     1$              ;DEVICE IS POINTED TO.
002120' 140067  176223                  BICB    R0,DONFLG       ;CLEAR THIS FLAG
002124' 104400  000000'                 EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
```



```
                                        ;SUBROUTINES
                                        ;-----------

                                ;THE FIRST IS THE VECTOR LOADING SUBROUTINE. THE VECTOR ADDRESS IS PASSED IN R0,
                                ;THE DEVICE ADDRESS IS PASSED IN R2. THE LINKING TABLE ADDRESS IS PASSED IN R3.
                                ;THE BUS REQUEST PRIORITY LEVEL IS A PARAMETER TAKEN INDIRECTLY THROUGH R5.

002130' 010320          VCTLOAD:MOV     R3,(R0)+        ;LOAD THE ADDRESS OF THE LINKING INSTRUCTION
002132' 113520                  MOVB    @(R5)+,(R0)+    ;AND THE PRIORITY LEVEL INTO THE VECTOR
002134' 005200                  INC     R0              ;POINT RO TO THE NEXT VECTOR ADDRESS
002136' 022323                  CMP     (R3)+,(R3)+     ;POINT R3 TO THE CSR INSERT LOCATION
002140' 010245                  MOV     R2,-(R5)        ;LOAD THE DEVICE ADDRESS IN THE LINK TABLE
002142' 005713                  TST     (R5)+           ;ALIGN R3 FOR THE NEXT DEVICE
002144' 000205                  RTS     R5              ;RETURN TO CALLING BLOCK

                                ;THIS ROUTINE SAVES THE REGISTER VALUES IN THE ERROR QUEUE, A FIRST-IN,
                                ;FIRST-OUT DISCIPLINE WRAPAROUND BUFFER.

002146' 016705  002766          SAVREG: MOV     ERRQPI,R5       ;USE R5 FOR INDEXING CAPABILITY
002152' 010025                          MOV     R0,(R5)+        ;SAVE R0...
002154' 010125                          MOV     R1,(R5)+        ;SAVE R1...
002156' 010225                          MOV     R2,(R5)+        ;SAVE R2...
002160' 010325                          MOV     R3,(R5)+        ;SAVE R3...
002162' 010425                          MOV     R4,(R5)+        ;AND R4
002164' 022705  003016'                 CMP     #ERRQUE+60,R5   ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
002170' 103002                          BHIS    1$              ;IF NO, GO RELOAD THE POINTER
002172' 012705  002736'                 MOV     #ERRQUE,R5      ;RESET R5 TO THE QUE BEGINNING
002176' 010567  002736          1$:     MOV     R5,ERRQPI       ;SET THE ENTRY POINTER TO NEXT ENTRY POINT
002202' 000207                          RTS     PC              ;RETURN TO THE CALLING BLOCK

                                ;THIS ROUTINE RETRIEVES REGISTER VALUES FROM THE ERROR QUEUE

002204' 016705  002732          GETREG: MOV     ERRQPO,R5       ;USE R5 FOR INDEXING CAPABILITY
002210' 012500                          MOV     (R5)+,R0        ;RETRIEVE R0...
002212' 012501                          MOV     (R5)+,R1        ;       R1...
002214' 012502                          MOV     (R5)+,R2        ;       R2...
002216' 012503                          MOV     (R5)+,R3        ;       R3...
002220' 012504                          MOV     (R5)+,R4        ;AND R4
002222' 022705  003016'                 CMP     #ERRQUE+60,R5   ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
002226' 103002                          BHIS    1$              ;IF NO, GO RELOAD THE POINTER
002230' 012705  002736'                 MOV     #ERRQUE,R5      ;RESET R5 TO THE QUEUE BEGINNING
002234' 010567  002702          1$:     MOV     R5,ERRQPO       ;SET THE RETRIEVAL POINTER TO THE NEXT FETCH POINT.
002240' 000207                          RTS     PC              ;RETURN TO THE CALLING BLOCK

                                ;THIS ROUTINE CALCULATES HOW MANY CHARACTERS ARE BEING TRANSMITTED IN EACH PATTE

002242' 116702  176100          GETLINE:MOVB    SLCTLIN,R2      ;COPY THE LINES SELECTED PARAMETER
002246' 005067  175762                  CLR     XMTCNT          ;RESET THE TOTAL TRANSMISSION COUNT
002252' 106002                  1$:     RORB    R2              ;GET A SELECTION BIT INTO THE CARRY BIT
002254' 005567  175754                  ADC     XMTCNT          ;ADD IT TO TOTAL NUMBER OF ACTIVE LINES
002260' 105702                          TSTB    R2              ;ARE 8. LINES CHECKED?
002262' 001373                          BEQ     1$              ;IF NOT, GO CHECK THE REST
002264' 006367  175744                  ASL     XMTCNT          ;NOW MULTIPLY BY 8.
002270' 006367  175740                  ASL     XMTCNT          ; DO THIS BY SHIFTING THREE TIMES
002274' 006367  175734                  ASL     XMTCNT          ;WE NOW KNOW HOW MANY CHARACTERS ARE IN EACH BURST
002300' 000207                          RTS     PC              ;RETURN TO THE CALLING SEGMENT
```

```
                                ;THIS ROUTINE CALCULATES THE BAUD RATE FROM THE SR1 OPTION. IF SR1 IS 0,THE DEFAULT
                                ;RATE OF 9600. IS ASSIGNED. THE LEAST SIGNIFICANT BIT IS THE ONLY ONE CONSIDERED.
                                ;THE BAUD RATE IS THEN LOADED INTO THE LINE PARAMETER WORD.

002302' 005767  175510          BAUDRTE:TST     SR1             ;IS A SPECIFIC RATE SELECTED?
002306' 001414                          BEQ     3$              ;IF NO,GO ASSIGN THE DEFAULT RATE
002310' 016704  175502                  MOV     SR1,R4          ;IF YES COPY SR1
002314' 012703  000015                  MOV     #15,R3          ;SET UP THE BIT CONFIGURATION FOR A BAUD RATE SELECTION
002320' 006004                  1$:     ROR     R4              ;ISOLATE THE NEXT BIT IN THE CARRY BIT
002322' 103403                          BCS     2$              ;IF THIS IS THE BIT, PUT TOGETHER THE BAUD RATE
002324' 005303                          DEC     R3              ;IF NOT,CALCULATE THE NEXT BIT CONFIGURATION
002326' 100413                          BMI     4$              ;IF NONE OF THE BITS WERE SET, RETURN TO THE CALLING PRO
002330' 000773                          BR      1$              ;GO CHECK THE NEXT ALTERNATIVE
002332' 150367  175667          2$:     BISB    R3,LINPAR+1     ;PLACE THE BAUD RATE IN THE LINE PARAMETER REGISTER
002336' 000207                          RTS     PC              ;RETURN TO THE CALLING PROCEDURE
002340' 052767  007000  175656  3$:     BIS     #7000,LINPAR    ;SET THE DEFAULT RATE(9600. BAUD)
002346' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
002352' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
002356' 000207                  4$:     RTS     PC              ;RETURN TO CALLING PROCEDURE

                                ;THIS ROUTINE DETERMINES WHAT TYPE OF ERROR WAS INDICATED BY THE SILO AND REPORTS EACH

002360' 010067  175754          STATERR:MOV     R0,RCVWORD      ;COPY THE INFORMATION RECEIVED
002364' 011167  175510                  MOV     (R1),CSRA       ;GET THE CONTENTS OF THE CONTROL REGISTER
002370' 010167  175506                  MOV     R1,ACSR         ;REPORT THE CSR CONTENTS
002374' 004767  177546                  JSR     PC,SAVREG       ;SAVE THE REGISTERS
002400' 005067  175502                  CLR     ERRTYP          ;UNKNOWN ERROR
                                        ;*****************************************************************
002404' 104405  000000' 005154'         ARDER$,BEGIN,TAB2       ;DEVICE RECEIVER ERROR
                                        ;*****************************************************************
002412' 004767  177566                  JSR     PC,GETREG       ;RESTORE THE REGISTERS
002416' 000207                          RTS     PC              ;RETURN TO CALLING PROCEDURE

                                ;THIS ROUTINE CALCULATES THE BUFFER ADDRESSES OF THE INCORRECT DATA AND REPORTS THE ERRO

002420' 004767  177522          DERROR: JSR     PC,SAVREG       ;SAVE THE PRESENT REGISTER VALUES
002424' 005741                          TST     -(R1)           ;POINT TO THE CSR ADDRESS AGAIN
002426' 010167  175446                  MOV     R1,CSRA         ;SHOW WHICH DEVICE IT WAS
002432' 116067  005120' 175446          MOVB    LNCKBF(R0),ASB  ;LOAD THE CORRECT DATA VALUE
002440' 116767  175704  175442          MOVB    RCVDATA,AGAS    ;LOAD THE ACTUAL DATA VALUE
002446' 005742                          TST     -(R2)           ;GET THE SILO ADDRESS OF THE DATA
002450' 010267  175430                  MOV     R2,WASADR       ;LOAD IT FOR REPORTING
002454' 062700  005120'                 ADD     #LNCKBF,R0      ;CALCULATE THE CORRECT VALUE ADDRESS
002460' 010067  175416                  MOV     R0,SBADR        ;REPORT IT
                                        ;*****************************************************************
002464' 104404  000000'                 DATER$,BEGIN            ;DATA ERROR!!!
                                        ;*****************************************************************
002470' 004767  177510                  JSR     PC,GETREG       ;RESTORE THE REGISTERS
002474' 000207                          RTS     PC              ;RETURN TO THE CALLING PROCEDURE

                                ;LINKAGE TABLE
                                ;EACH ENTRY CONSISTS OF A JSR INSTRUCTION FOLLOWED BY A LOCATION INTO WHICH THE CSR IS
                                ;LOADED, FOLLOWED BY THE SILO BUFFER ADDRESS, FOLLOWED BY A JSR FOR THE TRANSMITTER
                                ;INTERRUPT, THE CSR ADDRESS WORD, AND THE NUMBER OF THE INTERRUPTING DEVICE
```



```
002476' 004567  177016          LNKTAB: JSR     R5,RCVINT       ;LINK FOR RECEIVER 0
002502' 000000                          0
002504' 003120'                         SILO0
002506' 004567  176472                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 0
002512' 000000                          0
002514' 000000                          0
002516' 004567  176776                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 1
002522' 000000                          0
002524' 003320'                         SILO1
002526' 004567  176452                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 1
002532' 000000                          0
002534' 000001                          1
002536' 004567  176756                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 2
002542' 000000                          0
002544' 003520'                         SILO2
002546' 004567  176432                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 2
002552' 000000                          0
002554' 000002                          2
002556' 004567  176736                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 3
002562' 000000                          0
002564' 003720'                         SILO3
002566' 004567  176412                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 3
002572' 000000                          0
002574' 000003                          3
002576' 004567  176716                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 4
002602' 000000                          0
002604' 004120'                         SILO4
002606' 004567  176372                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 4
002612' 000000                          0
002614' 000004                          4
002616' 004567  176676                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 5
002622' 000000                          0
002624' 004320'                         SILO5
002626' 004567  176352                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 5
002632' 000000                          0
002634' 000005                          5
002636' 004567  176656                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 6
002642' 000000                          0
002644' 004520'                         SILO6
002646' 004567  176332                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 6
002652' 000000                          0
002654' 000006                          6
002656' 004567  176636                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 7
002662' 000000                          0
002664' 004720'                         SILO7
002666' 004567  176312                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 7
002672' 000000                          0
002674' 000007                          7

                                ;THESE ARE THE QUEUES.

002676' 000010                  XMTQUE: .BLKW   8.              ;TRANSMITTER SERVICE QUEUE
002716' 000010                  RCVQUE: .BLKW   8.              ;RECEIVER SERVICE QUEUE
002736' 000031                  ERRQUE: .BLKW   25.             ;ERROR SERVICE QUEUE

                                ;THESE ARE THE BUFFERS
```

```
                                XMTBUF: .BLKB   64.             ;TRANSMITTER BUFFERS, ONE CHARACTER FOR EACH LINE
                                SILO0:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 0
                                SILO1:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 1
                                SILO2:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 2
                                SILO3:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 3
                                SILO4:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 4
                                SILO5:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 5
                                SILO6:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 6
                                SILO7:  .BLKW   64.             ;RECEIVER BUFFER FOR DEVICE 7
                                LNCKBF: .BLKB   8.              ;BUFFER FOR CHECKING INDIVIDUAL LINE DATA

                                ;THESE ARE THE QUEUE POINTERS

                                XMTQPI: OPEN                    ;TRANSMITTER QUEUE ENTRY (IN) POINTER
                                XMTQPO: OPEN                    ;TRANSMITTER QUEUE RETRIEVAL POINTER (OUT)
                                RCVQPI: OPEN                    ;RECEIVER QUEUE POINTERS
                                RCVQPO: OPEN
                                ERRQPI: OPEN                    ;ERROR QUE POINTERS
                                ERRQPO: OPEN
                                NUMBA1: .WORD   0

                                ;THESE ARE THE ASCII MESSAGES AND ERROR PRINTOUT TABLES

                                TAB1:   XMTCNT                  ;TABLE 1 IS USED TO REPORT XMTCNTING CHARACTERS
                                                                ;FIRST, THE NUMBER TRANSMITTED IS GIVEN
                                        MISS                    ;THEN THE NUMBER OF CHARACTERS NOT RECEIVED
                                        177777                  ;TERMINATOR FOR TABLE 1
                                TAB2:   DVCNMBR                 ;TABLE 2 IS USED TO REPORT RECEIVER ERRORS
                                        RCVWORD                 ;WORD CONTAINING THE ERRORS
                                        177777                  ;TERMINATOR FOR TABLE 2
                                HUNG:   M1
                                        177777

005166' 042504  044526  042503  M1:     .ASCII  "DEVICE "
005174' 020040
005176' 000006                  M2:     .BLKB   6
005204' 044040  047125  000107  M3:     .ASCIZ  " HUNG"
                                        .EVEN
        000001                  .END
```



```
ACSR    000102R          389#    637*    741*    872*
ACTVAT  000642R          514#    609
ADDR    000006R          355#    469     516     548
ADDR22= 001000           407#
ASB     000106R          393#    886*
ASTAT   000104R          391#
AWAS    000110R          394#    887*
BAUDRT  002302R          525     852#
BEGIN   000000R          352#    458     569     570     593     597     602     607     645     649     652     709     713
                         714     735     736     746     789     864     865     876     893
BGNDAT  000336R          429#    538*    539*    756
BIT0    = 000001         407#
BIT1    = 000002         407#
BIT10   = 002000         407#
BIT11   = 004000         407#
BIT12   = 010000         407#    416     418
BIT13   = 020000         407#
BIT14   = 040000         407#    417
BIT15   = 100000         407#
BIT2    = 000004         407#
BIT3    = 000010         407#    412     419
BIT4    = 000020         407#    413     419
BIT5    = 000040         407#    414
BIT6    = 000100         407#    415
BIT7    = 000200         407#
BIT8    = 000400         407#
BIT9    = 001000         407#
BREAKS= 104407           407#    569     570     713     714     735     736     864     865
BR1     000012R          357#    484
BR2     000013R          358#    481
BTOD$ = 104421           407#
CDATA$= 104412           407#
CKDATA  001764R          732     754#
CONFIG  000056R          347#
CSRA    000100R          387#    636*    740*    871*    885*
DATA    000344R          432#    453*    537     546*
DATCKS= 104411           407#
DATERS= 104404           407#    893
DCLR    = 000020         413#    522     782
DERROR  002420R          772     883#
DONFLG  000347R          435#    515#    571     579     788*
DROP    000442R          457#    466
DVCNMB  000232R          426#    760*    786*    987
DVID1   000014R          359#    444     454
EIGHT   = 000030         419#    524
ENDITS= 104413           407#    607
ENDS    = 104410         407#    458     602
ERRQPI  005140R          497#    809     816*    977#
ERRQPO  005142R          498*    823     827*    978#
ERRQUE  002736R          497     468     816     829     831     956#
ERRTYP  000106R          392#    647*    744*    874*
EXITS = 104400           407#    652     789
FINISH  001174R          573     600     606#
GETLIN  002242R          549     837#
GETPAS= 104415           407#
GETREG  002204R          598     651     737     748     823#    878     895
```

```
GWBUFS= 104414          407#
HRDCNT  000044R         372#
HRDERS= 104405          407#    649     746     876
HRDPAS  000050R         374#
HUNG    005162R         597     990#
ICONT   000036R         369#
ICOUNT  000040R         370#
IDNUM   000122R         399#
INIT    000030R         366#
INITIA  000740R         519     536#
INTR    000120R         398#    451#
LCKOUT  000351R         437#    504*    711     716*    781*
LINPAR  000224R         423#    524*    527     530*    861*    863*
LNCKBF  005140R         765     770     773*    886     890     969#
LNKTAB  002476R         470     903#
LNXMCT  002236R         428#    505     666*    669*
LPR     = 000002        408#    527*
MAINT   = 000010        412#    526
MAP22S= 104416          407#
MISS    000342R         431#    742*    985
MODNAM  000000R         353#
MODSP   000224R         367     405#
MSENAB= 000040          414#    556
MSGNS   = 104403        407#    597
MSGSS   = 104402        407#
MSGS    = 104401        407#
MSR     = 000006        410#
M1      005166R         990     993#
M2      005176R         593     995#
M3      005204R         996#
NULL    = 000000        407#    649
NUMBAI  005144R         583     593     979#
OPEN    = 000000        354     360     361     362     363     380     381     382     383     384     385     386     387
                        389     391     393     394     396     397     398     407#    423     424     425     426     427
                        429     430     431     432     433     435     436     437#    973     974     975     976     977
                        978
OTOAS   = 104420        407#    593
PASCNT  000034R         368#
PIRQS   = 000004        407#    645     709
POPSP   = 005726        407#
POPSP2= 022626          407#
PRTY    = 000000        407#
PRTY0   = 000000        407#
PRTY1   = 000040        407#
PRTY2   = 000100        407#
PRTY3   = 000140        407#
PRTY4   = 000200        407#
PRTY5   = 000240        357     358     407#
PRTY6   = 000300        407#
PRTY7   = 000340        407#
PS      = 177776        407#
PSW     = 177776        407#
PUSH    = 005746        407#
PUSH2   = 024646        407#
RANDS   = 104417        407#
RANNUM  000054R         376#
```



```
RBUF    = 000002        407#
RCVDAT  000350R         436#    767*    770     887
RCVDON  002074R         763     781#
RCVINT  001520R         700#    903     909     915     921     927     933     939     945
RCVON   = 010000        418#    524
RCVQPI  005134R         495*    700*    703*    704     706*    975#
RCVQPO  005136R         496*    717     718*    719     721*    976#
RCVQUE  002716R         495     496     704     706     719     721     955#
RCVSRV  001564R         709     711#    715
RCVTMR  000230R         425#    726*    738*
RCVWDR  000340R         430#    870*    988
RESTRT  000446R         395     456     464#
RES1    000056R         378#
RES2    000060R         379#
RIE     = 000100        415#    677     701
RSTRT   000112R         395#
SAVREG  002146R         598     638     734     743     809#    873     883
SBADR   000102R         388#    891*
SELECT  000345R         433#    454*    465     514     547     581*    599
SETUP1  000446R         465#
SETUP2  000534R         474     491#
SILOEN= 010000          416#
SILO0   003120R         905     961#
SILO1   003320R         911     962#
SILO2   003520R         917     963#
SILO3   003720R         923     964#
SILO4   004120R         929     965#
SILO5   004320R         935     966#
SILO6   004520R         941     967#
SILO7   004720R         947     968#
SLCTLI  000346R         434#    557     837
SOFCNT  000042R         371#
SOFERS= 104406          407#
SOFPAS  000046R         373#
SPOINT  000032R         367#
SPSIZ   =       1#      400
SR1     000016R         360#    852     854
SR2     000020R         361#
SR3     000022R         362#
SR4     000024R         363#
START   000352R         366     442#
STAT    000026R         365#
STATER  002360R         766     870#
SVR0    000062R         380#
SVR1    000064R         381#
SVR2    000066R         382#
SVR3    000070R         383#
SVR4    000072R         384#
SVR5    000074R         385#
SVR6    000076R         386#
SYSCNT  000052R         375#
TAB1    005146R         746     983#
TAB2    005154R         872     987#
TCR     = 000004        409#    557*    675*
TDR     = 000006        411#    661*
TIE     = 040000        417#    556     660     678     680
```

```
TIMER   001056R           567#    577
TMRCNT  000226R           424#    566*    576*
TMRSET  001050R           553     566#
TRPDFD= 000022            407#
VCTLOA  002130R           490     483     798#
VECTOR  000010R           356#    468
WASADR  000104R           390#    889*
WDFR    000116R           397#    443*    450*
WDTO    000114R           396#    442*    449*
XPLAG   000005R           354#
XMTBUF  003020R           536     661     663*    960#
XMTCNT  000234R           427#    727     761     838#    840*    843*    844*    845*    983
XMTINT  001204R           617#    906     912     918     924     930     936     942     948
XMTQPI  005130R           491#    617*    618*    619     621*    973#
XMTQPO  005132R           493*    628     629*    630     632*    974#
XMTQUE  002676R           491     493     499     619     621     630     632     954#
XMTSRV  001250R           628#
.     = 005212R           428#    954#    955#    956#    960#    961#    962#    963#    964#    965#    966#    967#    968#
                          969#    995#

. ABS.  000000      000
        005212      001


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 XDZAC0,XDZAC0/SOL/CRF:SYM=DDXCOM,XDZAC0
 RUN-TIME:  2 3 .3 SECONDS
 RUN-TIME RATIO: 29/5=5.0
 CORE USED:  7K  (13 PAGES)
```