```
.REM _
```

IDENTIFICATION

PRODUCT CODE: AC-E911C-MC

PRODUCT NAME: CXDZBC0 DZV11 MODULE

PRODUCT DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

1. ABSTRACT:

DZB IS AN IOMOD THAT EXERCISES UP TO EIGHT (CONSECUTIVELY ADDRESSED) DZV11 ASYNCHRONOUS INTERFACES. IT USES MAINTENANCE MODE TO TRANSMIT AND RECEIVE A BINARY COUNT PATTERN OUTPUTTED AND RECEIVED IN 32. 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 FOUR LINES ARE RUN ON EACH SELECTED DEVICE.

2. REQUIREMENTS:

HARDWARE: AT LEAST ONE DZV11 INTERFACE; NO WRAPAROUND OR MAINTENANCE CABLE IS NEEDED

STORAGE:: DZB REQUIRES:
1. DECIMAL WORDS: 1021
2. OCTAL WORDS: 01775
3. OCTAL BYTES: 3772

3. PASS DEFINITION:

ONE PASS OF THE DZB MODULE CONSISTS OF TRANSMITTING AND RECEIVING 8960. CHARACTERS FOR EACH LINE OF EACH DZV11 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 LSI-11 AT 9600. BAUD.

5. CONFIGURATION PARAMETERS:

DEFAULT PARAMETERS:

DVA: 1, VCT: 1, BR1: 4, BR2: 4, DVC: 1, SR1: 0

REQUIRED PARAMETERS:

AT CONFIGURATION TIME THE USER MUST SPECIFY:

DVA:	ADDRESS OF FIRST DZV11 CSR REG.
VCT:	VECTOR ADDRESS OF FIRST DZV11
DVC:	NO OF DZV11'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 DZV11'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 DZV11 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 DZV11 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 DZV11'S SELECTED GENERATED BOTH TRANSMIT AND RECEIVE INTERRUPTS, THE APPROPRIATE BIT IN DONFLG FOR THAT DZV11 WILL BE CLEARED. IF THIS DOES NOT OCCUR IN THE GIVEN TIME, THE DEVICE NUMBER OF THE OFFENDING DZV11 WILL BE CALCULATED, AND THIS WILL BE REPORTED IN A MODULE MESSAGE. THE OFFENDING DEVICE IS DROPPED FROM THE EXERCISE. IF NO MORE DZV11'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 DZV11'S WERE SUCCESSFULLY EXERCISED OR IF MORE DZV11'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 DZV11 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. SERVICE IS DEFERRED TO LEVEL ZERO PRIORITY.

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 DZV11 IS ENABLED.

RCVINT: THIS SEGMENT SAVES A POINTER TO THE CSR ADDRESS OF THE INTERRUPTING DZV11 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 DZV11 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 DZV11'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 276=
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

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

MODULE LOCATION ICONT MAY BE MODIFIED TO VARY THE PERIOD BETWEEN END OF PASS REPORTS.

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

9. NON-STANDARD PRINTOUTS:

WHEN A STATUS ERROR IS DETECTED, DZB 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.

```
                ;       .TITLE  DZBC DEC/X11 SYSTEM EXERCISER MODULE
                ;       DDXCOM  VERSION  6      23-MAY-78
                                .LIST   BIN
                        ;*************************************************************************
        000000  000000          BEGIN:
        000000  055104  041502      040 MODNAM: .ASCII  /DZBC / ;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      200         BR1:    .BYTE   PRTY4+0         ;1ST BR LEVEL.
        000013      200         BR2:    .BYTE   PRTY4+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  000314          INIT:   START           ;MODULE START ADDR.
        000032  000224          SPOINT: MODSP           ;MODULE STACK POINTER.
        000034  000000          PASCNT: 0               ;PASS COUNTER.
        000036  002140          ICONT:  2140            ;# OF ITERATIONS PER PASS=2140
        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  000314          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  000142          IDNUM:  142             ;MODULE IDENTIFICATION NUMBER=142
        000224                  MODSP:
                        ;*************************************************************************
   390          000002                          RBUF=2          ;DEFINITION OF MODE 6 OFFSET TO THE READ BUFFER
```



```
   391          000002                          LPR=2           ;MODE 6 OFFSET TO THE LINE PARAMETER REGISTER
   392          000004                          TCR=4           ;OFFSET TO THE TRANSMITTER CONTROL REGISTER
   393          000006                          MSR=6           ;OFFSET TO THE MODEM STATUS REGISTER
   394          000006                          TDR=6           ;OFFSET TO THE TRANSMITTER DATA REGISTER
   395          000010                          MAINT=BIT3      ;ENABLE MAINTENANCE MODE BIT
   396          000020                          DCLR=BIT4       ;DEVICE MASTER CLEAR
   397          000040                          MSENAB=BIT5     ;ENABLE THE MASTER SCANNER (DEVICE GO)
   398          000100                          RIE=BIT6        ;RECEIVER INTERRUPT ENABLE
   399          010000                          SILOEN=BIT12    ;SILO ALARM INTERRUPT ENABLE
   400          040000                          TIE=BIT14       ;TRANSMITTER INTERRUPT ENABLE
   401          010000                          RCVON=BIT12     ;TURN THE RECEIVER ON (RECEIVER GO)
   402          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' 000040          LNXMCT: .BLKB   32.             ;TRANSMISSION COUNTERS FOR EACH DEVICE
000276' 000000          BGNDATA:OPEN                    ;TWO COPIES OF THE FIRST CHARACTER IN TRANSMIT PATTERN
000300' 000000          RCVWORD:OPEN                    ;USED TO REPORT RECEIVER ERROR
000302' 000000          MISS:   OPEN                    ;USED TO REPORT NUMBER OF CHARACTERS MISSING
000304'     000         DATA:   .BYTE   OPEN            ;CHARACTER BUILDING BYTE
000305'     000         SELECT: .BYTE   OPEN            ;ACTIVE DEVICE SELECTION PARAMETER
000306'     017         SLCTLIN:.BYTE   17              ;ACTIVE LINE SELECTION PARAMETER
000307'     000         DONFLG: .BYTE   OPEN            ;WATCHDOG FLAG FOR BUSY DEVICES
000310'     000         RCVDATA:.BYTE   OPEN            ;BUFFER FOR CHARACTER CHECKING
000311'     000         LCKOUT: .BYTE   OPEN            ;BUFFER ACCESS FLAG
000312' 000000          TEMP1:  .WORD   0               ;TEMPORARY STORAGE FOR ASCII CONVERSION
                                .EVEN
```



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

000314' 105067  177764  START:  CLRB    DATA            ;INITIALIZE THE DATA PATTERN WORD
000320' 116767  177470  177757  MOVB    DVID1,SELECT    ;COPY THE DEVICE SELECTION PARAMETER. ARE
                                                        ;ANY DEVICES SELECTED?
000326' 001002                  BNE     RESTRT          ;IF SO, BEGIN PROCESSING. IF NOT, DROP THE MODULE
000330'                 DROP:
000330' 104410  000000'         END$,BEGIN              ;DROP THE MODULE

                                        ;INITIALIZE THE NUMBER OF PASSES TO BE MADE. SET UP THE DZV11 INTERRUPT
                                        ;VECTORS TO INTERRUPT IN THE SUBROUTINE LINKING TABLE. CALL SUBROUTINE
                                        ;VCTLOAD FOR THIS PURPOSE.

000334'                 RESTRT:
000334' 116701  177745  SETUP1: MOVB    SELECT,R1       ;COPY THE DEVICE SELECTION PARAMETER INTO R1
000340' 001773                  BEQ     DROP            ;IF NO DEVICES SELECTED (ALL FLAGS CLEAR) DROP THE
                                                        ;MODULE
000342' 016700  177442          MOV     VECTOR,R0       ;LOAD THE VECTOR ADDRESS IN R0
000346' 016702  177434          MOV     ADDR,R2         ;LOAD R2 WITH ADDRESS OF FIRST DZ11
000352' 012703  002324'         MOV     #LNKTAB,R3      ;POINT R3 TO THE BEGINNING OF JSR LINKING TABLE
000356' 000241                  CLC                     ;MAKE SURE CARRY BIT IS CLEAR BEFORE ROTATION
000360' 106001          1$:     RORB    R1              ;ISOLATE A SELECTION FLAG IN THE CARRY BIT
000362' 103410                  BCS     3$              ;IF THE FLAG IS SET, GO SET UP THE VECTORS
000364' 001416                  BEQ     SETUP2          ;IF NO FLAGS ARE LEFT, GO SET UP THE BUFFERS
000366' 062700  000010          ADD     #10,R0          ;IF MORE FLAGS ARE SET, ADJUST POINTERS. THE
                                                        ;VECTOR POINTER...
000372' 062703  000020          ADD     #20,R3          ;THE LINKAGE TABLE POINTER...
000376' 062702  000010  2$:     ADD     #10,R2          ;AND THE ADDRESS POINTER
000402' 000766                  BR      1$              ;GO SET UP THE NEXT DZV11 ADDRESSES
000404' 004567  001344  3$:     JSR     R5,VCTLOAD      ;CALL THE VECTOR SETUP ROUTINE FOR RECEIVER
000410' 000013'                 BR2                     ;VECTOR, PASSING THE RECEIVER BR LEVEL AS THE
                                                        ;ARGUMENT
000412' 004567  001336          JSR     R5,VCTLOAD      ;SET UP THE TRANSMITTER VECTOR PASSING THE
000416' 000012'                 BR1                     ;TRANSMITTER BR LEVEL AS THE ARGUMENT
000420' 000766                  BR      2$              ;GO SETUP THE NEXT DZV11

                                        ;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.

000422' 012767  002524' 003262  SETUP2: MOV     #XMTQUE,XMTQPI  ;POINT THE TRANSMIT QUEUE ENTRY (IN) POINTER
                                                        ;TO THE BEGINNING OF THE QUEUE
000430' 012767  002524' 003256          MOV     #XMTQUE,XMTQPO  ;POINT THE TRANSMIT QUEUE RETRIEVAL (OUT)
                                                        ;POINTER TO THE BEGINNING OF THE QUEUE
000436' 012767  002544' 003252          MOV     #RCVQUE,RCVQPI  ;SET UP THE RECEIVER QUEUE POINTERS
000444' 012767  002544' 003246          MOV     #RCVQUE,RCVQPO  ;
000452' 012767  002564' 003242          MOV     #ERRQUE,ERRQPI  ;SETUP THE ERROR QUEUE POINTERS
000460' 012767  002564' 003236          MOV     #ERRQUE,ERRQPO  ;
000466' 012703  002524'                 MOV     #XMTQUE,R3      ;POINT R3 TO THE BEGINNING OF THE QUEUE AREA
000472' 012704  000473                  MOV     #315.,R4        ;USING R4 AS A COUNTER CLEAR -
000476' 005023          1$:     CLR     (R3)+           ;TRANSMITTER, RECEIVERS, AND ERROR QUEUES....
000500' 005304                  DEC     R4              ;TRANSMITTER TEXT BUFFER...
000502' 001375                  BNE     1$              ;RECEIVER SILO BUFFERS
000504' 105067  177601          CLRB    LCKOUT          ;CLEAR THE BUFFER ACCESS FLAG
000510' 012703  000236'         MOV     #LNXMCT,R3      ;POINT TO THE CHRACTER COUNTER FOR EACH LINE
```

```
                                ;SET UP 32. BYTE POINTERS
000514' 012704  000040          MOV     #32.,R4
000520' 112723  000010  2$:     MOVB    #0,(R3)+        ;SET UP THIS BYTE AND POINT TO THE NEXT ONE
000524' 005304                  DEC     R4              ;REDUCE THE COUNT. ARE 32. BYTES SET UP?
000526' 001374                  BNE     2$              ;IF NO, GO SET UP THE NEXT ONE

                                ;THIS BLOCK SETS THE WATCHDOG TIMER FLAG, INITIALIZES THE DZV11 CONDITIONS, SETS THE CHA
                                ;AND BAUD RATE FOR THE ACTIVE DZV11S

000530' 116700  177551  ACTVATE:MOVB    SELECT,R0       ;COPY THE DEVICE SELECTION PARAMETER
000534' 110067  177547          MOVB    R0,DONFLG       ;SET THE WATCHDOG TIMER DEVICE COMPLETION FLAG
000540' 016701  177242          MOV     ADDR,R1         ;LOAD THE ADDRESS OF THE FIRST DZV11
000544' 106000          1$:     RORB    R0              ;ISOLATE A SELECTION FLAG IN THE CARRY BIT
000546' 103404                  BCS     3$              ;IF SELECTED, GO SET UP THE DEVICE REGISTERS
000550' 001426                  BEQ     INITIAL         ;IF NO MORE SELECTED, GO SET UP THE DATA
000552' 062701  000010  2$:     ADD     #10,R1          ;POINT R1 TO THE NEXT DZV11
000556' 000772                  BR      1$              ;PROCESS NEXT DZV11
000560' 012711  000020  3$:     MOV     #DCLR,(R1)      ;MASTER CLEAR THIS DZV11
000564' 012702  000004          MOV     #4,R2           ;USE R2 TO COUNT 4. LINES BEING SET UP
000570' 012767  010030  177426  MOV     #RCVON!EIGHT,LINPAR ;SET UP NO PARITY, EIGHT BITS/CHAR.
000576' 004767  001324          JSR     PC,BAUDRTE      ;GO CALCULATE THE BAUD RATE TO BE USED
000602' 052711  000010          BIS     #MAINT,(R1)     ;ENABLE MAINTENANCE MODE OPERATION
000606' 016761  177412  000002  4$:     MOV     LINPAR,LPR(R1) ;SET UP THIS LINE'S PARAMETERS
000614' 005302                  DEC     R2              ;REDUCE THE COUNT. ARE ALL 4. LINES SET?
000616' 001755                  BEQ     2$              ;IF YES, GO PROCESS NEXT DZV11
000620' 005267  177400          INC     LINPAR          ;IF NO, INCREMENT THE LINE SELECTION PARAMETER
000624' 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

000626' 012701  002646' INITIAL:MOV     #XMTBUF,R1      ;POINT R1 TO THE START OF THE BUFFER AREA
000632' 116700  177446          MOVB    DATA,R0         ;BEGIN SETTING UP THE CHARACTER PATTERN
000636' 110067  177434          MOVB    R0,BGNDATA      ;COPY THE FIRST DATUM BEING TRANSMITTED IN THIS
000642' 110067  177431          MOVB    R0,BGNDATA+1    ;GROUP .BGNDATA WILL BE USED TO SET
                                                        ;UP LNCKBF(THE LINE CHECK BUFFER ) IN THE CKDATA ROUTINE
000646' 012702  000040          MOV     #32.,R2         ;USE R2 TO COUNT THE LOOP ITERATIONS
000652' 110021          1$:     MOVB    R0,(R1)+        ;PUT A CHARACTER IN THE BUFFER
000654' 005302                  DEC     R2              ;REDUCE COUNT. HAVE ALL CHARACTERS BEEN MADE?
000656' 001375                  BNE     1$              ;IF NO, GO LOAD THE NEXT CHARACTER
000660' 062700  000010          ADD     #8.,R0          ;POINT TO THE BEGINNING OF THE NEXT PATTERN
000664' 110067  177414          MOVB    R0,DATA         ;SAVE THAT PATTERN BEGINNING
000670' 116700  177411          MOVB    SELECT,R0       ;COPY THE DEVICE SELECTION PARAMETER
000674' 016701  177106          MOV     ADDR,R1         ;LOAD THE ADDRESS OF THE FIRST DZV11
000700' 004767  001162          JSR     PC,GETLINE      ;FIND OUT HOW MANY LINES ARE RUNNING
000704' 000241                  CLC                     ;MAKE SURE CARRY BIT IS CLEAR BEFORE ROTATION
000706' 106000          2$:     RORB    R0              ;ISOLATE A SELECTION FLAG IN THE CARRY BIT
000710' 103404                  BCS     4$              ;IF SELECTED, GO START THE DEVICE
000712' 001411                  BEQ     TMRSET          ;IF NO MORE SELECTED, GO START WATCHDOG TIMER
000714' 062701  000010  3$:     ADD     #10,R1          ;POINT R1 TO THE NEXT DZV11
000720' 000772                  BR      2$              ;GO PROCESS THE NEXT DZV11
000722' 052711  040040  4$:     BIS     #TIE!MSENAB,(R1) ;ENABLE TRANSMITTER INTERRUPTS
000726' 116761  177354  000004  MOVB    SLCTLIN,TCR(R1) ;ENABLE THE TRANSMITTERS FOR ALL SELECTED LINES
000734' 000767                  BR      3$              ;PROCESS THE NEXT DZV11
```



```
                                ;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.

000736' 012767  000005  177262  TMRSET: MOV     #5,TMRCNT       ;SET THE TIMER COUNT FOR THE BREAK LOOP
000744' 005004          TIMER:  CLR     R4              ;USING R4, RETURN TO MONITOR 65535 TIMES
000746'                 1$:
000746' 104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR...
000752' 104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
000756' 105767  177325          TSTB    DONFLG          ;IF DONFLG IS CLEAR, EACH SELECTED DEVICE WAS
                                                        ;SUCCESSFUL
000762' 001437                  BEQ     FINISH          ;IF SO, PERFORM ENDPASS PROCESSING
000764' 005304                  DEC     R4              ;IF NOT, REDUCE COUNT AND BREAK AGAIN
000766' 001367                  BNE     1$              ;BREAK IF COUNT NOT EXCEEDED
000770' 005367  177232          DEC     TMRCNT          ;REDUCE THE OVERALL TIME.IS IT EXCEEDED?
000774' 001363                  BNE     TIMER           ;IF NO,START ANOTHER BREAK LOOP

000776' 116703  177305          MOVB    DONFLG,R3       ;IF TIMEOUT OCCURRED, SAVE THE REMAINING FLAG
                                                        ;IN R3.
001002' 140367  177277          BICB    R3,SELECT       ;CLEAR THE SELECTION FLAG FOR THIS DEVICE
001006' 006003          2$:     ROR     R3              ;DETERMINE WHICH DEVICE WAS READ FOR REPORTING
                                                        ;PURPOSES
001010' 103402                  BCS     3$              ;IF THIS IS THE DEVICE, R4 CONTAINS THE CORRECT
                                                        ;LINE NUMBER... GO REPORT IT
001012' 005204                  INC     R4              ;IF NOT, INCREMENT R4, WHICH WAS INITIALLY 0
                                                        ;FROM THE PREVIOUS LOOP
001014' 000774                  BR      2$              ;GO SEE IF THIS IS THE DEVICE
001016' 010467  177270  3$:     MOV     R4,TEMP1        ;THE DEVICE NUMBER WILL BE REPORTED AS STATC
                                ;*****************************************************************
                                                        ;CONVERT TEMP1 TO ASCII AND
001022' 104420  000000' 000312' OTOA$,BEGIN,TEMP1,M2    ;STORE AT M2
001030' 003756'
                                ;*****************************************************************
001032' 004767  000734          JSR     PC,SAVREG       ;SAVE THE REGISTERS BEFORE LEAVING THE MODULE
001036' 104403  000000' 003742' MSGN$,BEGIN,HUNG         ;ASCII MESSAGE CALL WITH COMMON HEADER
001044' 004767  000760          JSR     PC,GETREG       ;RESTORE THE PREVIOUS REGISTER VALUES
001050' 105767  177231          TSTB    SELECT          ;ARE THERE ANY DEVICES STILL ACTIVE?
001054' 001002                  BNE     FINISH          ;IF YES, REDUCE COUNT AND CONTINUE
                                                        ;IF NO, DROP THE MODULE
001056' 104410  000000'         END$,BEGIN              ;DROP THE MODULE

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

001062'                 FINISH:
001062' 104413  000000'         ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                        ;MONITOR SHALL TEST END OF PASS
001066' 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.
```

```
                                                
001072' 010577  002614          XMTINT: MOV     R5,@XMTQPI      ;LOAD THE OFFSET TO THE CSR INTO TRANSMITTER QUEUE
001076' 062767  000002  002606          ADD     #2,XMTQPI       ;UPDATE THE QUEUE ENTRY POINTER
001104' 027767  002542' 002600          CMP     #XMTQUE+16,XMTQPI ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001112' 103003                          BHIS    1$              ;IF NO, CONTINUE PROCESSING
001114' 012767  002524' 002570          MOV     #XMTQUE,XMTQPI  ;IF YES, RESET THE POINTER TO QUEUE BEGINNING
001122' 012605                  1$:     MOV     (SP)+,R5        ;RESTORE THE PREVIOUS R5 VALUE
                                        ;-------------------------------------------------------------
001124' 000004  000000' 001132'         PIRQ$,BEGIN,XMTSRV              ; QUEUE UP TO CONTINUE AT XMTSRV AND RTI
                                        ;-------------------------------------------------------------
001132' 017701  002556          XMTSRV: MOV     @XMTQPO,R1      ;FETCH THE OFFSET FROM THE QUEUE
001136' 062767  000002  002550          ADD     #2,XMTQPO       ;UPDATE THE QUEUE RETRIEVAL POINTER
001144' 027767  002542' 002542          CMP     #XMTQUE+16,XMTQPO ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001152' 103003                          BHIS    1$              ;IF NO, CONTINUE PROCESSING
001154' 012767  002524' 002532          MOV     #XMTQUE,XMTQPO  ;IF YES, RESET POINTER TO BEGINNING OF THE QUEUE
001162' 011100                  1$:     MOV     (R1),R0         ;LOAD CSR ADDRESS INTO R0
001164' 005710                          TST     (R0)            ;IS THIS A VALID INTERRUPT?
001166' 100420                          BMI     2$              ;IF YES, GO ENABLE RECEIVER INTERRUPT
001170' 010067  176704                  MOV     R0,CSRA         ;IF NO, SET UP THE CSR ADDRESS FOR ERROR REPORTING
001174' 011067  176702                  MOV     (R0),ACSR       ;SET UP THE CONTENTS
001200' 004767  000566                  JSR     PC,SAVREG       ;SAVE THE REGISTERS BEFORE ERROR MESSAGE
001204' 012767  000011  176674          MOV     #11,ERRTYP
                                        ;*************************************************************
001212' 104405  000000' 000000          ERDER$,BEGIN,NULL       ;FALSE INTERRUPT REQUEST FROM TRANSMITTER
                                        ;*************************************************************
001220' 004767  000604                  JSR     PC,GETREG       ;RESTORE THE REGISTERS
001224' 104400  000000'                 EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
001230' 116002  000001          2$:     MOVB    1(R0),R2        ;GET THE LINE NUMBER FROM THE HIGH BYTE OF CSR
001234' 042702  177774                  BIC     #177774,R2      ;ISOLATE THE LINE NUMBER IN R2
001240' 016103  000002                  MOV     2(R1),R3        ;GET THE NUMBER OF THIS DEVICE
001244' 006303                          ASL     R3              ;MULTIPLY THE DEVICE NUMBER BY FOUR
001246' 006303                          ASL     R3              ;THIS INVOLVES MULTIPLYING IT BY TWO TWICE
001250' 060203                          ADD     R2,R3           ;CALCULATE THIS LINE'S TABLE OFFSET
001252' 042710  040000                  BIC     #TIE,(R0)       ;DISABLE INTERRUPTS BEFORE AN OVERRUN OCCURS
001256' 105763  000236'                 TSTB    LNXMCT(R3)      ;ALL CHARAC. TRANSMITTED?
001262' 001017                          BNE     5$              ;IF NOT GO LOAD BUFFER
001264' 112763  000010  000236'         MOVB    #8.,LNXMCT(R3)  ;IF SO, RESET THE COUNT OF CHARACTERS
001272' 012703  000001                  MOV     #1,R3           ;SET A BIT POINTER IN R3
001276' 005302                  3$:     DEC     R2              ;USE R2 AS A COUNT OF ROTATIONS TO MAKE
001300' 100402                          BMI     4$              ;IS THE POINTER NOW AT THE RIGHT LINE?
001302' 006303                          ASL     R3              ;IF NOT, POINT IT TO THE NEXT LINE
001304' 000774                          BR      3$              ;GO SEE IF THIS IS THE LINE
001306' 040360  000004          4$:     BIC     R3,TCR(R0)      ;IF SO, DISABLE TRANSMISSIONS ON THIS LINE
001312' 001012                          BNE     6$              ;IF MORE TRANSMITTERS ARE ACTIVE, GO EXIT
001314' 052710  000100                  BIS     #RIE,(R0)       ;OTHERWISE, ENABLE THE RECEIVER
001320' 000411                          BR      7$              ;SKIP RESETTING THE INTERRUPT ENABLE
001322' 116360  002646' 000006  5$:     MOVB    XMTBUF(R3),TDR(R0)      ;LOAD THE CHAR INTO TRANS BUFFER
001330' 105263  002646'                 INCB    XMTBUF(R3)      ;CREATE NEXT CHARACTER
001334' 105363  000236'                 DECB    LNXMCT(R3)      ;COUNT A CHARAC. FOR THIS LINE
001340' 052710  040000          6$:     BIS     #TIE,(R0)       ;RESTART TRANS. INTERRUPTS
001344'                         7$:
001344' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
```



```
                                        ;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 IS CHECKED, IT RELEASES
                                ;THE LINE CHECK BUFFER AND CLEAR THE CORRESPONDING BIT IN DONFLG.

001350' 010577  002342          RCVINT: MOV     R5,@RCVQPI      ;STORE THE OFFSET IN THE RETRIEVAL QUEUE
001354' 042735  000100                  BIC     #RIE,@(R5)+     ;DISABLE THIS RECEIVER'S INTERRUPTS BEFORE
                                                                ;ITS OTHER LINES OVERRUN THE QUEUES
001360' 062767  000002  002330          ADD     #2,RCVQPI       ;UPDATE THE QUEUE ENTRY POINTER
001366' 022767  002562' 002322          CMP     #RCVQUE+16,RCVQPI ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001374' 103003                          BHIS    1$              ;IF NOT, CONTINUE PROCESSING
001376' 012767  002544' 002312          MOV     #RCVQUE,RCVQPI  ;RESET THE POINTER TO QUEUE BEGINNING
001404' 012605                  1$:     MOV     (SP)+,R5        ;RESTORE THE PREVIOUS VALUE OF R5
                                        ;-------------------------------------------------------------
001406' 000004  000000' 001414'         PIRQ$,BEGIN,RCVSRV              ; QUEUE UP TO CONTINUE AT RCVSRV AND RTI
                                        ;-------------------------------------------------------------
001414' 105767  176671          RCVSRV: TSTB    LCKOUT          ;TEST THE BUFFER LOCK FLAG. IS ACCESS PERMITTED?
001420' 001405                          BEQ     11$             ;IF YES, GO SET THE LOCK AND CHECK THE DATA
001422' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001426' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001432' 000770                          BR      RCVSRV          ;TRY TO ACCESS AGAIN
001434' 105167  176651          11$:    COMB    LCKOUT          ;DENY OTHER ACCESSES TO THE BUFFER
001440' 017700  002254                  MOV     @RCVQPO,R0      ;FETCH THE OFFSET FROM THE QUEUE
001444' 062767  000002  002246          ADD     #2,RCVQPO       ;UPDATE THE QUEUE RETRIEVAL POINTER
001452' 022767  002562' 002240          CMP     #RCVQUE+16,RCVQPO ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
001460' 103003                          BHIS    1$              ;IF NO, CONTINUE PROCESSING
001462' 012767  002544' 002230          MOV     #RCVQUE,RCVQPO  ;IF YES, RESET THE POINTER TO THE QUEUE BEGINNING
001470' 012001                  1$:     MOV     (R0)+,R1        ;LOAD THE CSR ADDRESS INTO R1
001472' 042721  030300                  BIC     #30300,(R1)+    ;DISABLE RECEIVER INTERRUPT AND INTERRUPT ENABLE
                                                                ;AND POINT R1 TO THE NEXT RECEIVED CHAR. REGISTER
001476' 011002                          MOV     (R0),R2         ;LOAD THE SOFTWARE SILO BUFFER ADDRESS
001500' 012767  002000  176522          MOV     #2000,RCVTMR    ;SET THE RECEIVER BREAK LOOP TIMER
001506' 016703  176522                  MOV     XMTCNT,R3       ;SET THE COUNT FOR STORING THE DATA
001512' 111122                  2$:     MOVB    (R1),(R2)+      ;STORE A WORD IN THE SOFTWARE SILO
001514' 002003                          BGE     3$              ;IF THE DATA WASN'T VALID, GO ALLOW A LITTLE TIME FOR IT
001516' 005303                          DEC     R3              ;IF IT WAS VALID, REDUCE THE COUNT.
001520' 001374                          BNE     2$              ;IF NOT ZERO, GO STORE THE NEXT WORD
001522' 000434                          BR      CKDATA          ;IF ALL CHARACTERS RECEIVED, GO CHECK THEM
001524' 005742                  3$:     TST     -(R2)           ;RESET R2 TO THE PROPER BUFFER LOCATION
001526' 004767  000240                  JSR     PC,SAVREG       ;SAVE THE REGISTER VALUES
001532' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001536' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001542' 004767  000262                  JSR     PC,GETREG       ;RESTORE THE REGISTER VALUES
001546' 005367  176456                  DEC     RCVTMR          ;HAS ENOUGH TIME BEEN PERMITTED FOR A CHARACTER?
001552' 100357                          BPL     2$              ;IF NO, TRY TO GET ONE.
001554' 014067  176320                  MOV     -(R0),CSRA      ;LOAD THE DEVICE ADDRESS
001560' 013067  176316                  MOV     @(R0)+,ACSR     ;LOAD THE CSR CONTENTS
001564' 010367  176512                  MOV     R3,MISS         ;REPORT THE NUMBER OF CHARACTERS MISSING
001570' 004767  000176                  JSR     PC,SAVREG       ;SAVE THE REGISTERS
```

```
001574' 012767  000014  176304          MOV     #14,ERRTYP
                                        ;*************************************************************
001602' 104405  000000' 003726'         ARDER$,BEGIN,TAB1       ;DID NOT RECEIVE ALL TRANSMITTED CHARACTERS
                                        ;*************************************************************
001610' 004767  000214                  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.

001614' 012705  003706'         CKDATA: MOV     #LNCKBF,R5      ;POINT TO THE BEGINNING OF THE BUFFER
001620' 016725  176452                  MOV     BGNDATA,(R5)+   ;SET UP FOUR LINE CHECK BYTES
001624' 016725  176446                  MOV     BGNDATA,(R5)+   ;
001630' 011002                          MOV     (R0),R2         ;LOAD THE SOFTWARE SILO ADDRESS IN R2
001632' 016067  000010  176372          MOV     10(R0),DVCNMBR  ;SAVE THE NUMBER OF THIS DEVICE
001640' 016703  176370                  MOV     XMTCNT,R3       ;LOAD COUNT TO COMPARE CHARACTERS
001644' 012200                  3$:     MOV     (R2)+,R0        ;FETCH A WORD FROM THE SILO INTO R0
001646' 002024                          BGE     RCVDONE         ;IF IT'S INVALID, GO CLEAN UP BUFFERS
001650' 032700  070000                  BIT     #70000,R0       ;ARE THERE ANY TRANSMISSION ERRORS?
001654' 001402                          BEQ     4$              ;IF NO, GO DETERMINE THE LINE NUMBER
001656' 004767  000322                  JSR     PC,STATERR      ;IF YES, GO DETERMINE THE ERROR TYPE
001662' 110067  176422          4$:     MOVB    R0,RCVDATA      ;SAVE THE RECEIVED CHARACTER
001666' 000300                          SWAB    R0              ;PUT THE LINE NUMBER IN R0'S LOWER BYTE
001670' 042700  177774                  BIC     #177774,R0      ;ISOLATE THE LINE NUMBER IN R0
001674' 126067  003706' 176406          CMPB    LNCKBF(R0),RCVDATA ;IS THE DATA CORRECT?
001702' 001402                          BEQ     5$              ;IF YES, GO CHECK THE NEXT CHARACTER
001704' 004767  000336                  JSR     PC,DERROR       ;IF NO, GO REPORT A DATA ERROR
001710' 105260  003706'         5$:     INCB    LNCKBF(R0)      ;SET UP THE NEXT CHARACTER FOR THIS LINE
001714' 005303                          DEC     R3              ;REDUCE THE COUNT, ARE ALL CHARACTERS CHECKED?
001716' 001352                          BNE     3$              ;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.

001720' 105067  176365          RCVDONE:CLRB    LCKOUT          ;RESET THE BUFFER ACCESS SEMAPHORE
001724' 012741  000020                  MOV     #DCLR,-(R1)     ;CLEAR THIS DEVICE AND DISABLE IT
001730' 005000                          CLR     R0              ;THE DELETION FLAG WILL PROPAGATE IN R0
001732' 000261                          SEC                     ;USE THE CARRY BIT TO BUILD THE DELETION FLAG
001734' 006100                  1$:     ROL     R0              ;MOVE THE FLAG TO THE NEXT DEVICE
001736' 005367  176270                  DEC     DVCNMBR         ;WHEN THIS NUMBER IS NEGATIVE, THE CORRECT
001742' 002374                          BGE     1$              ;DEVICE IS POINTED TO.
001744' 140067  176337                  BICB    R0,DONFLG       ;CLEAR THIS FLAG
001750' 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.

001754' 010320          VCTLOAD:MOV     R3,(R0)+        ;LOAD THE ADDRESS OF THE LINKING INSTRUCTION
001756' 113520                  MOVB    @(R5)+,(R0)+    ;AND THE PRIORITY LEVEL INTO THE VECTOR
001760' 005200                  INC     R0              ;POINT R0 TO THE NEXT VECTOR ADDRESS
001762' 022323                  CMP     (R3)+,(R3)+     ;POINT R3 TO THE CSR INSERT LOCATION
001764' 010223                  MOV     R2,(R3)+        ;LOAD THE DEVICE ADDRESS IN THE LINK TABLE
001766' 005723                  TST     (R3)+           ;ALIGN R3 FOR THE NEXT DEVICE
001770' 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.

001772' 016705  001724          SAVREG: MOV     ERRQPI,R5       ;USE R5 FOR INDEXING CAPABILITY
001776' 010025                  MOV     R0,(R5)+        ;SAVE R0...
002000' 010125                  MOV     R1,(R5)+        ;SAVE R1...
002002' 010225                  MOV     R2,(R5)+        ;SAVE R2...
002004' 010325                  MOV     R3,(R5)+        ;SAVE R3...
002006' 010425                  MOV     R4,(R5)+        ;AND R4
002010' 022705  002644'         CMP     #ERRQUE+60,R5   ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
002014' 103002                  BHIS    1$              ;IF NO, GO RELOAD THE POINTER
002016' 012705  002564'         MOV     #ERRQUE,R5      ;RESET R5 TO THE QUE BEGINNING
002022' 010567  001674          1$:     MOV     R5,ERRQPI       ;SET THE ENTRY POINTER TO NEXT ENTRY POINT
002026' 000207                  RTS     PC              ;RETURN TO THE CALLING BLOCK

                                ;THIS ROUTINE RETRIEVES REGISTER VALUES FROM THE ERROR QUEUE

002030' 016705  001670          GETREG: MOV     ERRQPO,R5       ;USE R5 FOR INDEXING CAPABILITY
002034' 012500                  MOV     (R5)+,R0        ;RETRIEVE R0...
002036' 012501                  MOV     (R5)+,R1        ;       R1...
002040' 012502                  MOV     (R5)+,R2        ;       R2...
002042' 012503                  MOV     (R5)+,R3        ;       R3...
002044' 012504                  MOV     (R5)+,R4        ;AND R4
002046' 022705  002644'         CMP     #ERRQUE+60,R5   ;HAS THE QUEUE BOUNDARY BEEN EXCEEDED?
002052' 103002                  BHIS    1$              ;IF NO, GO RELOAD THE POINTER
002054' 012705  002564'         MOV     #ERRQUE,R5      ;RESET R5 TO THE QUEUE BEGINNING
002060' 010567  001640          1$:     MOV     R5,ERRQPO       ;SET THE RETRIEVAL POINTER TO THE NEXT FETCH POINT.
002064' 000207                  RTS     PC              ;RETURN TO THE CALLING BLOCK

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

002066' 116702  176214          GETLINE:MOVB    SLCTLIN,R2      ;COPY THE LINES SELECTED PARAMETER
002072' 005067  176136                  CLR     XMTCNT          ;RESET THE TOTAL TRANSMISSION COUNT
002076' 106002                  1$:     RORB    R2              ;GET A SELECTION BIT INTO THE CARRY BIT
002100' 005567  176130                  ADC     XMTCNT          ;ADD IT TO TOTAL NUMBER OF ACTIVE LINES
002104' 105702                          TSTB    R2              ;ARE 4. LINES CHECKED?
002106' 001373                          BNE     1$              ;IF NOT, GO CHECK THE REST
002110' 006367  176120                  ASL     XMTCNT          ;NOW MULTIPLY BY 8.
002114' 006367  176114                  ASL     XMTCNT          ; DO THIS BY SHIFTING THREE TIMES
002120' 006367  176110                  ASL     XMTCNT          ;WE NOW KNOW HOW MANY CHARACTERS ARE IN EACH BURST
002124' 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.

002126' 005767  175664          BAUDRTE:TST     SR1             ;IS A SPECIFIC RATE SELECTED?
002132' 001414                          BEQ     3$              ;IF NO,GO ASSIGN THE DEFAULT RATE
002134' 016704  175656                  MOV     SR1,R4          ;IF YES COPY SR1
002140' 012703  000015                  MOV     #15,R3          ;SET UP THE BIT CONFIGURATION FOR A BAUD RATE SELECTION
002144' 006004                  1$:     ROR     R4              ;ISOLATE THE NEXT BIT IN THE CARRY BIT
002146' 103403                          BCS     2$              ;IF THIS IS THE BIT, PUT TOGETHER THE BAUD RATE
002150' 005303                          DEC     R3              ;IF NOT,CALCULATE THE NEXT BIT CONFIGURATION
002152' 100413                          BMI     4$              ;IF NONE OF THE BITS WERE SET, RETURN TO THE CALLING PRO
002154' 000773                          BR      1$              ;GO CHECK THE NEXT ALTERNATIVE
002156' 150367  176043          2$:     BISB    R3,LINPAR+1     ;PLACE THE BAUD RATE IN THE LINE PARAMETER REGISTER
002162' 000207                          RTS     PC              ;RETURN TO THE CALLING PROCEDURE
002164' 052767  007000  176032  3$:     BIS     #7000,LINPAR    ;SET THE DEFAULT RATE(9600. BAUD)
002172' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
002176' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
002202' 000207                  4$:     RTS     PC              ;RETURN TO CALLING PROCEDURE

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

002204' 010067  176070          STATERR:MOV     R0,RCVWORD      ;COPY THE INFORMATION RECEIVED
002210' 011167  175664                  MOV     (R1),CSRA       ;GET THE CONTENTS OF THE CONTROL REGISTER
002214' 010167  175662                  MOV     R1,ACSR         ;REPORT THE CSR CONTENTS
002220' 004767  177546                  JSR     PC,SAVREG       ;SAVE THE REGISTERS
002224' 012767  000017  175654          MOV     #17,ERRTYP
                                        ;*****************************************************************
002232' 104405  000000' 003734'         HRDER$,BEGIN,TAB2       ;DEVICE RECEIVER ERROR
                                        ;*****************************************************************
002240' 004767  177564                  JSR     PC,GETREG       ;RESTORE THE REGISTERS
002244' 000207                          RTS     PC              ;RETURN TO CALLING PROCEDURE

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

002246' 004767  177520          DERROR: JSR     PC,SAVREG       ;SAVE THE PRESENT REGISTER VALUES
002252' 005741                          TST     -(R1)           ;POINT TO THE CSR ADDRESS AGAIN
002254' 010167  175620                  MOV     R1,CSRA         ;SHOW WHICH DEVICE IT WAS
002260' 116067  003706' 175620          MOVB    LNCKBF(R0),ASB  ;LOAD THE CORRECT DATA VALUE
002266' 116767  176016  175614          MOVB    RCVDATA,AWAS    ;LOAD THE ACTUAL DATA VALUE
002274' 005742                          TST     -(R2)           ;GET THE SILO ADDRESS OF THE DATA
002276' 010267  175602                  MOV     R2,WASADR       ;LOAD IT FOR REPORTING
002302' 062700  003706'                 ADD     #LNCKBF,R0      ;CALCULATE THE CORRECT VALUE ADDRESS
002306' 010067  175570                  MOV     R0,SBADR        ;REPORT IT
                                        ;*****************************************************************
002312' 104404  000000'                 DATER$,BEGIN            ;DATA ERROR!!!
                                        ;*****************************************************************
002316' 004767  177506                  JSR     PC,GETREG       ;RESTORE THE REGISTERS
002322' 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
```



```
002324' 004567  177020          LNKTAB: JSR     R5,RCVINT       ;LINK FOR RECEIVER 0
002330' 000000                          0
002332' 002706'                         SILO0
002334' 004567  176532                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 0
002340' 000000                          0
002342' 000000                          0
002344' 004567  177000                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 1
002350' 000000                          0
002352' 003006'                         SILO1
002354' 004567  176512                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 1
002360' 000000                          0
002362' 000001                          1
002364' 004567  176760                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 2
002370' 000000                          0
002372' 003106'                         SILO2
002374' 004567  176472                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 2
002400' 000000                          0
002402' 000002                          2
002404' 004567  176740                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 3
002410' 000000                          0
002412' 003206'                         SILO3
002414' 004567  176452                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 3
002420' 000000                          0
002422' 000003                          3
002424' 004567  176720                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 4
002430' 000000                          0
002432' 003306'                         SILO4
002434' 004567  176432                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 4
002440' 000000                          0
002442' 000004                          4
002444' 004567  176700                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 5
002450' 000000                          0
002452' 003406'                         SILO5
002454' 004567  176412                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 5
002460' 000000                          0
002462' 000005                          5
002464' 004567  176660                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 6
002470' 000000                          0
002472' 003506'                         SILO6
002474' 004567  176372                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 6
002500' 000000                          0
002502' 000006                          6
002504' 004567  176640                  JSR     R5,RCVINT       ;LINK FOR RECEIVER 7
002510' 000000                          0
002512' 003606'                         SILO7
002514' 004567  176352                  JSR     R5,XMTINT       ;LINK FOR TRANSMITTER 7
002520' 000000                          0
002522' 000007                          7

                                        ;THESE ARE THE QUEUES.

002524' 000010                  XMTQUE: .BLKW   8.              ;TRANSMITTER SERVICE QUEUE
002544' 000010                  RCVQUE: .BLKW   8.              ;RECEIVER SERVICE QUEUE
002564' 000031                  ERRQUE: .BLKW   25.             ;ERROR SERVICE QUEUE

                                        ;THESE ARE THE BUFFERS
```

```
                                        ;TRANSMITTER BUFFERS, ONE CHARACTER FOR EACH LINE
003646' 000040          XMTBUF: .BLKB   32.
003706' 000040          SILO0:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 0
003006' 000040          SILO1:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 1
003106' 000040          SILO2:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 2
003206' 000040          SILO3:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 3
003306' 000040          SILO4:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 4
003406' 000040          SILO5:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 5
003506' 000040          SILO6:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 6
003606' 000040          SILO7:  .BLKW   32.     ;RECEIVER BUFFER FOR DEVICE 7
003706' 000004          LNCKBF: .BLKB   4.      ;BUFFER FOR CHECKING INDIVIDUAL LINE DATA

                        ;THESE ARE THE QUEUE POINTERS

003712' 000000          XMTQPI: OPEN            ;TRANSMITTER QUEUE ENTRY (IN) POINTER
003714' 000000          XMTQPO: OPEN            ;TRANSMITTER QUEUE RETRIEVAL POINTER (OUT)
003716' 000000          RCVQPI: OPEN            ;RECEIVER QUEUE POINTERS
003720' 000000          RCVQPO: OPEN
003722' 000000          ERRQPI: OPEN            ;ERROR QUE POINTERS
003724' 000000          ERRQPO: OPEN

                        ;THESE ARE THE ASCII MESSAGES AND ERROR PRINTOUT TABLES

003726' 000234'         TAB1:   XMTCNT          ;TABLE 1 IS USED TO REPORT XMTCNTING CHARACTERS
                                                ;FIRST, THE NUMBER TRANSMITTED IS GIVEN
003730' 000302'                 MISS            ;THEN THE NUMBER OF CHARACTERS NOT RECEIVED
003732' 177777                  177777          ;TERMINATOR FOR TABLE 1
003734' 000232'         TAB2:   DVCNMBR         ;TABLE 2 IS USED TO REPORT RECEIVER ERRORS
003736' 000300'                 RCVWORD         ;WORD CONTAINING THE ERRORS
003740' 177777                  177777          ;TERMINATOR FOR TABLE 2
003742' 003746'         HUNG:   M1
003744' 177777                  177777

003746' 042504 044526 042503 M1:        .ASCII  'DEVICE  '
003754' 020040
003756' 000006          M2:     .BLKB   6
003764' 044040 047125 000107 M3:        .ASCIZ  ' HUNG'
                                .EVEN
000001                          .END
```



```
ACSR    000102R          377#    608*    694*    823*
ACTVAT  000530R          487#    582
ADDR    000006R          343#    442     489     521
ADDR22= 001000           390#
ASB     000106R          381#    837*
ASTAT   000104R          379#
AWAS    000110R          382#    838*
BAUDRT  002126R          498     803#
BEGIN   000000R          340#    431     542     543     566     570     575     580     597     612     615     640     662
                         666     667     688     689     699     740     815     816     827     844
BGNDAT  000276R          412#    511*    512*    708     709
BIT0  = 000001           390#
BIT1  = 000002           390#
BIT10 = 002000           390#
BIT11 = 004000           390#
BIT12 = 010000           390#    399     401
BIT13 = 020000           390#
BIT14 = 040000           390#    400
BIT15 = 100000           390#
BIT2  = 000004           390#
BIT3  = 000010           390#    395     402
BIT4  = 000020           390#    396     402
BIT5  = 000040           390#    397
BIT6  = 000100           390#    398
BIT7  = 000200           390#
BIT8  = 000400           390#
BIT9  = 001000           390#
BREAK$= 104407           390#    542     543     666     667     688     689     815     816
BR1     000012R          345#    457
BR2     000013R          346#    454
BTOD$ = 104421           390#
CDATA$= 104412           390#
CKDATA  001614R          685     707#
CONFIG  000056R          365#
CSRA    000100R          375#    607*    693*    822*    836*
DATA    000304R          415#    426*    510     519*
DATCK$= 104411           390#
DATER$= 104404           390#    844
DCLR  = 000020           396#    495     733
DERROR  002246R          723     834#
DONFLG  003078R          418#    488*    544     552     739*
DROP    000330R          430#    439
DVCNMB  000232R          409#    711*    737*    937
DVID1   000014R          347#    427
EIGHT = 000030           402#    497
ENDIT$= 104413           390#    580
ENDS  = 104410           390#    431     575
ERRQPI  003722R          470#    760*    928#
ERRQPO  003724R          471*    774     783*    929#
ERRQUE  002564R          470     471     766     768     780     782     907#
ERRTYP  000106R          380#    610*    697*    825*
EXIT$ = 104400           390#    615     640     740
FINISH  001062R          546     573     579#
GETLIN  002066R          522     788#
GETPA$= 104415           390#
GETREC  002030R          571     614     690     701     774#    829     846
```

```
GWBUF$= 104414          390#
HRDCNT  000044R         360#
HRDERS= 104405          390#    612     699     827
HRDPAS  000050R         362#
HUNG    003742R         570     940#
ICONT   000036R         357#
ICOUNT  000040R         358#
IDNUM   000122R         387#
INIT    000030R         354#
INITIA  000626R         492     509#
INTR    000120R         386#
LCKOUT  000311R         420#    477*    664     669*    732*
LINPAR  000224R         406#    497*    500     503*    812*    814*
LNCKBF  003706R         707     721     724*    837     841     920#
LNKTAB  002324R         443     854#
LNXMCT  000236R         411#    478     623     625*    637*
LPR   = 000002          391#    500*
MAINT = 000010          395#    499
MAP22$= 104416          390#
MISS    000302R         414#    695*    935
MODNAM  000000R         341#
MODSP   000224R         355     388#
MSENAB= 000040          397#    529
MSGN$ = 104403          390#    570
MSGS$ = 104402          390#
MSG$  = 104401          390#
MSR   = 000006          393#
M1      003746R         940     943#
M2      003756R         566     945#
M3      003764R         946#
NULL  = 000000          390#    612
OPEN  = 000000          342     348     349     350     351     368     369     370     371     372     373     374     375
                        377     379     381     382     384     385     386     390#    406     407     408     409     410
                        412     413     414     415     416     418     419     420     924     925     926     927     928
                        929
OTOA$ = 104420          390#    566
PASCNT  000034R         356#
PIRQ$ = 000004          390#    597     662
POPSP = 005726          390#
POPSP2= 022626          390#
PRTY  = 000000          390#
PRTY0 = 000000          390#
PRTY1 = 000040          390#
PRTY2 = 000100          390#
PRTY3 = 000140          390#
PRTY4 = 000200          345     346     390#
PRTY5 = 000240          390#
PRTY6 = 000300          390#
PRTY7 = 000340          390#
PS    = 177776          390#
PSW   = 177776          390#
PUSH  = 005746          390#
PUSH2 = 024646          390#
RAND$ = 104417          390#
RANNUM  000054R         364#
RBUF  = 000002          390#
```



```
RCVDAT  000310R         419#    718*    721     838
RCVDON  001720R         714     732#
RCVINT  001350R         653#    854     860     866     872     878     884     890     896
RCVON = 010000          401#    497
RCVQPI  003716R         468*    653*    656*    657     659*    926#
RCVQPO  003720R         469*    670     671*    672     674*    927#
RCVQUE  002544R         468     469     657     659     672     674     906#
RCVSRV  001414R         662     664#    668
RCVTMR  000730R         408#    679*    691*
RCVWOR  000300R         413#    821*    938
RESTRT  000334R         383     429     437#
RES1    000056R         366#
RES2    000060R         367#
RIE   = 000100          398#    633     654
RSTRT   000112R         383#
SAVREG  001772R         569     609     687     696     760#    824     834
SBADR   000102R         376#    842*
SELECT  000305R         416#    427*    438     487     520     554*    572
SETUP1  000334R         438#
SETUP2  000422R         447     464#
SILOEN= 010000          399#
SILO0   002706R         856     912#
SILO1   003006R         862     913#
SILO2   003106R         868     914#
SILO3   003206R         874     915#
SILO4   003306R         880     916#
SILO5   003406R         886     917#
SILO6   003506R         892     918#
SILO7   003606R         898     919#
SLCTLI  000306R         417#    530     788
SOFCNT  000042R         359#
SOFERS= 104406          390#
SOFPAS  000046R         361#
SPOINT  000032R         355#
SPSIZ = 000040            1#    388
SR1     000016R         348#    803     805
SR2     000020R         349#
SR3     000022R         350#
SR4     000024R         351#
START   000314R         354     426#
STAT    000026R         353#
STATER  002204R         717     821#
SVR0    000062R         368#
SVR1    000064R         369#
SVR2    000066R         370#
SVR3    000070R         371#
SVR4    000072R         372#
SVR5    000074R         373#
SVR6    000076R         374#
SYSCNT  000052R         363#
TAB1    003726R         699     933#
TAB2    003734R         827     937#
TCR   = 000004          392#    530*    631*
TDR   = 000006          394#    635*
TEMP1   000312R         421#    562*    566
TIE   = 040000          400#    529     622     638
```

```
TIMER   000744R           540#    550
TMRCNT  000226R           407#    539*    549*
TMRSET  000736R           526     539#
TRPDFD= 000022            390#
VCTLOA  001754R           453     456     749#
VECTOR  000010R           344#    441
WASADR  000104R           378#    840*
WDFR    000116R           385#
WDTO    000114R           384#
XFLAG   000005R           342#
XMTBUF  002546R           509     635     636*    911#
XMTCNT  000234R           410#    680     712     789*    791*    794*    795*    796*    933
XMTINT  001072R           590#    857     863     869     875     881     887     893     899
XMTQPI  003712R           464*    590*    591*    592     594*    924#
XMTQPO  003714R           466*    599     600*    601     603*    925#
XMTQUE  002524R           464     466     472     592     594     601     603     905#
XMTSRV  001132R           597     599#
.       = 003772R         411#    905#    906#    907#    911#    912#    913#    914#    915#    916#    917#    918#    919#
                          920#    945#

. ABS.  000000      000
        003772      001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 XDZBC0,XDZBC0/SQL/CRF:SYM=DDXCOM,XDZBC0
 RUN-TIME:  2  3  .4 SECONDS
 RUN-TIME RATIO: 19/5=3.2
 CORE USED:  7K  (13 PAGES)
```