.REM_

IDENTIFICATION

PRODUCT CODE: AC-E929B-MC

PRODUCT NAME: CXKWDB0 KW11-K MODULE

PRODUCT DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

1.0 ABSTRACT

THE KWD IS AN IOMOD THAT EXERCISES THE KW11K DUAL REAL TIME CLOCK. ON START IT EXERCISES THE CSRS AND PRESET BUFFERS OF BOTH CLOCKS. ON RESET AND AFTER ENDPASS, IT EXCERCISES EACH CLOCK SEPARATELY AND TOGETHER AT EACH OF THEIR BASIC RATES.

2.0 REQUIREMENTS

HARDWARE: ONE KW11-K

STORAGE:: KWD REQUIRES:
1. DECIMAL WORDS: 838
2. OCTAL WORDS: 1506
3. OCTAL BYTES: 3214

3.0 PASS DEFINITION

ONE PASS OF THE KWD MODULE CONSISTS OF GENERATING INTERRUPTS FOR ONE SECOND AT EACH CLOCKS RATES, TOGETHER AND SEPARATE, UNTIL 60 SECONDS HAVE ELAPSED.

4.0 EXECUTION TIME

ONE PASS OF THE KWD MODULE RUNNING ALONE TAKES APPROXIMATELY ONE MINUTE.

5.0 CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:

DEVADR: 170404, VECTOR 344, BR1: 6

DEVCNT: 1, SR1: 0

REQUIRED PARAMETERS:

NONE

6.0 DEVICE/OUTPUT SET-UP:

GROUND SCHMITT TRIGGER INPUTS #1,2,3

7.0 MODULE OPERATION

TEST SEQUENCE:

1. (START) BIT EXERCISE CSR,PRESET REGISTER OF CLOCK A.
2. BIT EXERCISE CSR,PRESET REGISTER OF CLOCK B.

   (RESTRT) COUNT TESTS USING INTERRUPTS COUNT INTERRUPTS WILL OCCUR IN ONE SECOND AND ADVANCE THE TEST TO THE NEXT RATE.

   AFTER A RATE HAS BEEN SELECTED, A CHECK IS MADE TO SEE IF THE OPERATOR HAS INHIBITED THAT RATE FROM TEST. IF NOT, CONTROL IS TRANSFERRED TO THE PARTICULAR RATE ROUTINE (LISTED BELOW). EACH RATE ROUTINE MUST PRELOAD THE BUFFER REGISTER OF CLOCKS A AND B TO THE COUNT THAT WILL CAUSE IT TO INTERRUPT IN ONE SECOND. AFTER THE BUFFER IS LOADED, THE CSR IS LOADED WITH THE PROPER BITS THAT SELECT THE RATE.

   CLOCK B INTERRUPTS ALMOST IMMEDIATELY SINCE ITS BUFFER REGISTER IS ONLY 8 BITS LONG AND CAN NOT HOLD A LARGE PRESET NUMBER. WHEN CLOCK A INTERRUPTS IT CHECKS TO SEE IF CLOCK B HAS INTERRUPTED IF NOT, IT REPORTS AN ERROR.

   A. COUNT TEST CLOCK A RATE 1MHZ.
      CLOCK B RATE 1 MHZ

   B. CLOCK A RATE: 100KHZ
      CLOCK B RATE: 100KHZ

   C. CLOCK A RATE: 10KHZ
      CLOCK B RATE: 10KHZ

   D. CLOCK A RATE: 1KHZ
      CLOCK B RATE: 1KHZ

   E. CLOCK A RATE: 100HZ
      CLOCK B RATE: 100HZ

   F. CLOCK A RATE: LINE FREQ.
      CLOCK B RATE: LINE FREQ.

   G. CLOCK A RATE: PSEUDO RANDOM (1 OF 3 RATES)
      CLOCK B RATE: PSEUDO RANDOM (1 OF 3 RATES)

   H. CLOCK A RATE: OVERFLOW CLOCK B

SEQ 0004

CLOCK B RATE: 1MHZ

8.0 OPERATION OPTIONS

VALID SR1 VALUES

SR1 BIT	ENABLE/DISABLE	FUNCTION
0	0	ENABLE TESTING 1MHZ
	1	DISABLE TESTING 1MHZ
1	0	ENABLE TESTING 100KHZ
	1	DISABLE TESTING 100KHZ
2	0	ENABLE TESTING 10KHZ
	1	DISABLE TESTING 10KHZ
3	0	ENABLE TESTING 1KHZ
	1	DISABLE TESTING 1KHZ
4	0	ENABLE TESTING 100HZ
	1	DISABLE TESTING 100HZ
5	0	*ENABLE TESTING RANDOM
	1	DISABLE TESTING RANDOM
6	0	ENABLE TESTING LINE FREQ
	1	DISABLE TESTING LINE FREQ
7	0	*ENABLE TESTING OVERFLOW B
	1	DISABLE TESTING OVERFLOW B

*NOTE: IF RANDOM RATE OR OVERFLOW B RATE IS SELECTED, THEN AN SR1 BIT DISABLING A PARTICULAR RATE WILL BE IGNORED.

9.0 NON-STANDARD PRINTOUTS:

ALL PRINTOUTS HAVE THE STANDARD FORMATS DESCRIBED IN THE DEC/X11 DOCUMENT.

-

```
                                        IOMOD   <KWDB >,170404,344,6,6,,60.,,102
                                        MODULE  140000,KWDB ,170404,344,6,6,,60.,,102
                                        .TITLE  KWDB DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6       23-MAY-78
                                        .LIST   BIN
                                ;*****************************************************************************
                                BEGIN:
000000' 053513  041104      040 MODNAM: .ASCII  /KWDB / ;MODULE NAME.
000005'     000 XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 170404  ADDR:   170404+0                ;1ST DEVICE ADDR.
000010' 000344  VECTOR: 344+0                   ;1ST DEVICE VECTOR.
000012'     300 BR1:    .BYTE   PRTY6+0         ;1ST BR LEVEL.
000013'     300 BR2:    .BYTE   PRTY6+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' 000266' INIT:   START                   ;MODULE START ADDR.
000032' 000224' SPOINT: MODSP                   ;MODULE STACK POINTER.
000034' 000000  PASCNT: 0                       ;PASS COUNTER.
000036' 000074  ICONT:  60.                     ;# OF ITERATIONS PER PASS=60.
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' 002014' 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' 000102  IDNUM:  102                     ;MODULE IDENTIFICATION NUMBER=102
        000040          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                        .NLIST
```



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

```
                                        ;MODULE REQUIRED REGISTERS - SET UP BY THIS MODULE.

000224' 170404                  ASR:    .WORD   170404          ;CLOCK A STATUS REG.
000226' 170406                  ABR:    .WORD   170406          ;CLOCK A BUFFER REG.
000230' 170430                  ACR:    .WORD   170430          ;CLOCK A COUNT REG.

000232' 170432                  BSR:    .WORD   170432          ;CLOCK B STATUS REG.
000234' 170434                  BBR:    .WORD   170434          ;CLOCK B BUFFER REG.
000236' 170436                  BCR:    .WORD   170436          ;CLOCK B COUNT REG.

000240' 000344                  AVECT:  .WORD   344             ;CLOCK A INTERRUPT VECTOR.
000242' 000346                  AVECT2: .WORD   346

000244' 000364                  BVECT:  .WORD   364             ;CLOCK B INTERRUPT VECTOR.
000246' 000366                  BVECT2: .WORD   366


000250' 000001                  RATEP:  .WORD   1               ;POINTS TO CURRENT RATE
000252' 000000                  OFF:    .WORD   0               ;OFFSET TO TAKE US TO RATE ROUTINE
000254' 000000                  RANA:   .WORD   0               ;RANDOM NUMBER.
000256' 000000                  RANB:   .WORD   0               ;RANDOM NUMBER.
000260' 000000                  AIFLG:  .WORD   0               ;FLAG TO SHOW THAT CLOCK A HAS INTERRRUPTED.
000262' 000000                  BIFLG:  .WORD   0               ;FLAG TO SHOW THAT CLOCK B HAS INTERRUPTED.
000264' 000000                  TRY:    0               ;*****************************************

000266' 012767  000010  177612  START:  MOV     #8.,ERRTYP              ;8 INTERRUPTS/ITERATION

000274' 016767  177506  177722          MOV     ADDR,ASR        ;GET BASE ADDR.
000302' 016767  177502  177730          MOV     VECTOR,AVECT    ;GET BASE VECTOR ADDR.

000310' 016700  177710                  MOV     ASR,R0          ;NOW WE'RE GONNA FIX
000314' 062700  000002                  ADD     #2,R0           ;ALL CLOCK ADDRESSES BASED ON ASR.
000320' 010067  177702                  MOV     R0,ABR
000324' 062700  000022                  ADD     #22,R0
000330' 010067  177674                  MOV     R0,ACR
000334' 062700  000002                  ADD     #2,R0
000340' 010067  177666                  MOV     R0,BSR
000344' 062700  000002                  ADD     #2,R0
000350' 010067  177660                  MOV     R0,BBR
000354' 062700  000002                  ADD     #2,R0
000360' 010067  177652                  MOV     R0,BCR

000364' 016700  177650                  MOV     AVECT,R0        ;NOW FIX VECTOR ADDRESSES
000370' 062700  000020                  ADD     #20,R0
000374' 010067  177644                  MOV     R0,BVECT
000400' 016767  177634  177634          MOV     AVECT,AVECT2
000406' 062767  000002  177626          ADD     #2,AVECT2
000414' 016767  177624  177624          MOV     BVECT,BVECT2
000422' 062767  000002  177616          ADD     #2,BVECT2

                                        ;*
                                        ;*LOGIC TEST #1 BE SURE A CLOCK EXISTS AT THE
                                        ;*SPECIFIED ADDR. IF NO CLOCK, THEN A
                                        ;*DEC/X11 SYS ERROR WILL OCCUR.
                                        ;*
```



```
000430' 005777  177570          LOG1:   TST     @ASR            ;ADDRESS THE CLOCK. IF SYS ERROR
                                                                ;OCCURS, THEN CLOCK DID NOT
                                                                ;RETURN SLAVE-SYN WHEN
                                                                ;ADDRESSED.

                                        ;*
                                        ;*LOGIC TEST #2. MAKE SURE CLOCK A CSR BITS
                                        ;*15,13,8,6,3, AND 1 CAN BE SET + CLEARED.
                                        ;*

000434' 012767  120512  177442  LOG2:   MOV     #120512,ASTAT   ;GENERATE + RECORD PATTERN TO BE USED.
000442' 016777  177436  177554          MOV     ASTAT,@ASR      ;SET THEM IN CSR OF CLOCK A.
000450' 017767  177550  177424          MOV     @ASR,ACSR       ;READ THEM BACK
000456' 026767  177422  177416          CMP     ASTAT,ACSR      ;DID THEY ALL SET?
000464' 001415                          BEQ     2$              ;YES - GO TO NEXT TEST.
000466' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
000472' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
000476'                         1$:
000476' 016767  177522  177374          MOV     ASR,CSRA        ;RECORD CSR'S ADDR
000504' 012767  000025  177374          MOV     #25,ERRTYP      ;BIT STUCK
                                        ;*****************************************************************
000512' 104405  000000' 000000          HRDER$,BEGIN,NULL       ;;PATTERN 120512 FAILED
                                        ;*****************************************************************

000520' 005077  177500          2$:     CLR     @ASR            ;TRY CLEARING THE BITS
000524' 017767  177474  177350          MOV     @ASR,ACSR       ;READ IT BACK.
000532' 001417                          BEQ     LOG3            ;IF ZERO CSR GOOD.
000534' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
000540' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
000544' 005067  177334          3$:     CLR     ASTAT           ;EXPECT ZERO CSR.
000550' 016767  177450  177322          MOV     ASR,CSRA        ;RECORD CSR'S ADDR.
000556' 012767  000025  177322          MOV     #25,ERRTYP      ;BIT STUCK
                                        ;*****************************************************************
000564' 104405  000000' 000000          HRDER$,BEGIN,NULL       ;;CSR FAILED TO CLEAR
                                        ;*****************************************************************


                                        ;*
                                        ;*LOGIC TEST #3. MAKE SURE CLOCK A CSR BITS
                                        ;*14,9,7,5,2, AND 0 CAN BE SET + CLEARED.
                                        ;*

000572' 012767  041245  177304  LOG3:   MOV     #41245,ASTAT    ;GENERATE + RECORD PATTERN TO BE USED.
000600' 016777  177300  177416          MOV     ASTAT,@ASR      ;SET THEM IN CSR OF CLOCK A.
000606' 017767  177412  177266          MOV     @ASR,ACSR       ;READ THEM BACK
000614' 026767  177264  177260          CMP     ASTAT,ACSR      ;DID THEY ALL SET?
000622' 001415                          BEQ     2$              ;YES - GO TO NEXT TEST.
000624' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
000630' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
000634'                         1$:
000634' 016767  177364  177236          MOV     ASR,CSRA        ;RECORD CSR'S ADDR.
000642' 012767  000025  177236          MOV     #25,ERRTYP      ;BIT STUCK
                                        ;*****************************************************************
000650' 104405  000000' 000000          HRDER$,BEGIN,NULL       ;;CSR PATTERN 41245 FAILED
                                        ;*****************************************************************
```

```
                                                    2$:     CLR     @ASR            ;TRY CLEARING THE BITS
000662' 017767  177336  177212                              MOV     @ASR,ACSR       ;READ IT BACK.
000670' 001417                                              BEQ     LOG4            ;IF ZERO CSR GOOD.
000672' 104407  000000'                                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
000676' 104407  000000'                                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
000702' 005067  177176                              3$:     CLR     ASTAT           ;EXPECT ZERO CSR.
000706' 016767  177312  177164                              MOV     ASR,CSRA        ;RECORD CSR'S ADDR.
000714' 012767  000025  177164                              MOV     #25,ERRTYP      ;BIT STUCK
                                                            ;*****************************************************************
000722' 104405  000000' 000000                              HRDER$,BEGIN,NULL       ;;CSR FAILED TO CLEAR
                                                            ;*****************************************************************


                                                            ;*
                                                            ;*LOGIC TEST #4. MAKE SURE CLOCK B CSR BITS
                                                            ;*11,6,4, AND 2 CAN BE SET + CLEARED.
                                                            ;*

000730' 012767  004124  177146  LOG4:      MOV     #4124,ASTAT     ;GENERATE + RECORD PATTERN TO BE USED.
000736' 016777  177142  177266              MOV     ASTAT,@BSR      ;SET THEM IN CSR OF CLOCK B.
000744' 017767  177262  177130              MOV     @BSR,ACSR       ;READ THEM BACK
000752' 026767  177126  177122              CMP     ASTAT,ACSR      ;DID THEY ALL SET?
000760' 001415                              BEQ     2$              ;YES - GO TO NEXT TEST.
000762' 104407  000000'                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
000766' 104407  000000'                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
000772'                             1$:
000772' 016767  177234  177100              MOV     BSR,CSRA        ;RECORD CSR'S ADDR.
001000' 012767  000025  177100              MOV     #25,ERRTYP      ;BIT STUCK
                                            ;*****************************************************************
001006' 104405  000000' 000000              HRDER$,BEGIN,NULL       ;;CSR PATTERN 4124 FAILED
                                            ;*****************************************************************


001014' 005077  177212              2$:     CLR     @BSR            ;TRY CLEARING THE BITS
001020' 017767  177206  177054              MOV     @BSR,ACSR       ;READ IT BACK.
001026' 001417                              BEQ     LOG5            ;IF ZERO CSR GOOD.
001030' 104407  000000'                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001034' 104407  000000'                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001040' 005067  177040              3$:     CLR     ASTAT           ;EXPECT ZERO CSR.
001044' 016767  177162  177026              MOV     BSR,CSRA        ;RECORD CSR ADDR.
001052' 012767  000025  177026              MOV     #25,ERRTYP      ;BIT STUCK
                                            ;*****************************************************************
001060' 104405  000000' 000000              HRDER$,BEGIN,NULL       ;;CSR FAILED TO CLEAR
                                            ;*****************************************************************


                                            ;*
                                            ;*LOGIC TEST #5. MAKE SURE CLOCK B CSR BITS
                                            ;*7,5,3,1, AND 0 CAN BE SET + CLEARED.
                                            ;*

001066' 012767  000253  177010  LOG5:      MOV     #253,ASTAT      ;GENERATE + RECORD PATTERN TO BE USED.
001074' 016777  177004  177130              MOV     ASTAT,@BSR      ;SET THEM IN CSR OF CLOCK B.
001102' 017767  177124  176772              MOV     @BSR,ACSR       ;READ THEM BACK
001110' 026767  176770  176764              CMP     ASTAT,ACSR      ;DID THEY ALL SET?
001116' 001415                              BEQ     2$              ;YES - GO TO NEXT TEST.
001120' 104407  000000'                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001124' 104407  000000'                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001130'                             1$:
```



```
001130' 016767  177076  176742              MOV     BSR,CSRA        ;RECORD CSR'S ADDR.
001136' 012767  000025  176742              MOV     #25,ERRTYP      ;BIT STUCK
                                            ;*****************************************************************
001144' 104405  000000' 000000              HRDER$,BEGIN,NULL       ;;CSR PATTERN 253 FAILED
                                            ;*****************************************************************


001152' 005077  177054              2$:     CLR     @BSR            ;TRY CLEARING THE BITS
001156' 017767  177050  176716              MOV     @BSR,ACSR       ;READ IT BACK.
001164' 001417                              BEQ     LOG6            ;IF ZERO CSR GOOD.
001166' 104407  000000'                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001172' 104407  000000'                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001176' 005067  176702              3$:     CLR     ASTAT           ;EXPECT ZERO CSR.
001202' 016767  177024  176670              MOV     BSR,CSRA        ;RECORD CSR'S ADDR.
001210' 012767  000025  176670              MOV     #25,ERRTYP      ;BIT STUCK
                                            ;*****************************************************************
001216' 104405  000000' 000000              HRDER$,BEGIN,NULL       ;;CSR FAILED TO CLEAR
                                            ;*****************************************************************


                                            ;*
                                            ;*LOGIC TEST #6. MAKE SURE CLOCK A BUFFER REG
                                            ;*PATTERN 125252 CAN BE SET + CLEARED.
                                            ;*

001224' 012767  125252  176652  LOG6:      MOV     #125252,ASTAT   ;GENERATE + RECORD PATTERN TO BE USED.
001232' 016777  176646  176766              MOV     ASTAT,@ABR      ;SET THEM IN BUFFER REG OF CLOCK A.
001240' 017767  176762  176634              MOV     @ABR,ACSR       ;READ THEM BACK
001246' 026767  176632  176626              CMP     ASTAT,ACSR      ;DID THEY ALL SET?
001254' 001415                              BEQ     2$              ;YES - GO TO NEXT TEST.
001256' 104407  000000'                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001262' 104407  000000'                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001266'                             1$:
001266' 016767  176734  176604              MOV     ABR,CSRA
001274' 012767  000025  176604              MOV     #25,ERRTYP      ;BIT STUCK
                                            ;*****************************************************************
001302' 104405  000000' 000000              HRDER$,BEGIN,NULL       ;;BUFFER REG PATTERN 125252 FAILED
                                            ;*****************************************************************


001310' 005077  176712              2$:     CLR     @ABR            ;TRY CLEARING THE BITS
001314' 017767  176706  176560              MOV     @ABR,ACSR       ;READ IT BACK.
001322' 001417                              BEQ     LOG7            ;IF ZERO BUFFER GOOD.
001324' 104407  000000'                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001330' 104407  000000'                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001334' 005067  176544              3$:     CLR     ASTAT           ;EXPECT ZERO BUFFER.
001340' 016767  176662  176532              MOV     ABR,CSRA        ;RECORD ADDR. OF BUFFER REG.
001346' 012767  000025  176532              MOV     #25,ERRTYP      ;BIT STUCK
                                            ;*****************************************************************
001354' 104405  000000' 000000              HRDER$,BEGIN,NULL       ;;BUFFER REG A FAILED TO CLEAR
                                            ;*****************************************************************


                                            ;*
                                            ;*LOGIC TEST #7. MAKE SURE CLOCK A BUFFER REG
                                            ;*PATTERN 052525 CAN BE SET + CLEARED.
                                            ;*

001362' 012767  052525  176514  LOG7:      MOV     #052525,ASTAT   ;GENERATE + RECORD PATTERN TO BE USED.
001370' 016777  176510  176630              MOV     ASTAT,@ABR      ;SET THEM IN BUFFER OF CLOCK A.
```

```
                                                ;READ THEM BACK
                                                ;DID THEY ALL SET?
                                                ;YES - GO TO NEXT TEST.
                                                ;TEMPORARY RETURN TO MONITOR....
                                                ;THEN CONTINUE AT NEXT INSTRUCTION.
                                1$:
                                                ;RECORD BUFFER REG ADDR.
                                                ;BIT STUCK
                                                ;*****************************************************************
                                                ;;BUFF REG PATTERN 052525 FAILED
                                                ;*****************************************************************
                                2$:
                                                ;TRY CLEARING THE BITS
                                                ;READ IT BACK.
                                                ;IF ZERO BUFFER GOOD.
                                                ;TEMPORARY RETURN TO MONITOR....
                                                ;THEN CONTINUE AT NEXT INSTRUCTION.
                                3$:
                                                ;EXPECT ZERO BUFFER.
                                                ;RECORD BUFFER REG A ADDR.
                                                ;BIT STUCK
                                                ;*****************************************************************
                                                ;;BUFFER REG FAILED TO CLEAR
                                                ;*****************************************************************

                                                ;*
                                                ;*LOGIC TEST #8. MAKE SURE CLOCK B BUFFER REG
                                                ;*PATTERN 252 CAN BE SET + CLEARED.
                                                ;*
LOG8:                                           ;GENERATE + RECORD PATTERN TO BE USED.
                                                ;SET THEM IN BUFFER OF CLOCK B.
                                                ;READ THEM BACK
                                                ;DID THEY ALL SET?
                                                ;YES - GO TO NEXT TEST.
                                                ;TEMPORARY RETURN TO MONITOR....
                                                ;THEN CONTINUE AT NEXT INSTRUCTION.
                                1$:
                                                ;RECORD BUFFER REG B ADDR
                                                ;BIT STUCK
                                                ;*****************************************************************
                                                ;;BUFFER REG PATTERN 252 FAILED
                                                ;*****************************************************************
                                2$:
                                                ;TRY CLEARING THE BITS
                                                ;READ IT BACK.
                                                ;IF ZERO BUFFER GOOD.
                                                ;TEMPORARY RETURN TO MONITOR....
                                                ;THEN CONTINUE AT NEXT INSTRUCTION.
                                3$:
                                                ;EXPECT ZERO BUFFER.
                                                ;RECORD BUFFER REG ADDR.
                                                ;BIT STUCK
                                                ;*****************************************************************
                                                ;;BUFFER REG FAILED TO CLEAR
                                                ;*****************************************************************

                                                ;*
```



```
                                                ;*LOGIC TEST #9. MAKE SURE CLOCK B BUFFER REG
                                                ;*PATTERN 125 CAN BE SET + CLEARED.
                                                ;*
LOG9:                                           ;GENERATE + RECORD PATTERN TO BE USED.
                                                ;SET THEM IN BUFFER OF CLOCK B.
                                                ;READ THEM BACK
                                                ;DID THEY ALL SET?
                                                ;YES - GO TO NEXT TEST.
                                                ;TEMPORARY RETURN TO MONITOR....
                                                ;THEN CONTINUE AT NEXT INSTRUCTION.
                                1$:
                                                ;RECORD BUFFER REG ADDR
                                                ;BIT STUCK
                                                ;*****************************************************************
                                                ;;BUFFER REG PATTERN 125 FAILED
                                                ;*****************************************************************
                                2$:
                                                ;TRY CLEARING THE BITS
                                                ;READ IT BACK.
                                                ;IF ZERO BUFFER GOOD.
                                                ;TEMPORARY RETURN TO MONITOR....
                                                ;THEN CONTINUE AT NEXT INSTRUCTION.
                                3$:
                                                ;EXPECT ZERO BUFFER.
                                                ;RECORD BUFFER REG ADDR.
                                                ;BIT STUCK
                                                ;*****************************************************************
                                                ;;BUFFER REG FAILED TO CLEAR
                                                ;*****************************************************************
RESTRT:                                         ;PRESET REANDOM NUMBER GENERATOR
                                                ;FIRST PASS THRU LOOP, OFFSET=0, RATES WILL=1.
                                                ;SET UP CLOCK A'S INTER. VECTOR.
                                                ;SET UP CLOCK B'S INTER. VECTOR.
                                                ;SET PRIORITY ON CLOCK A'S INTR.
                                                ;SET PRIORITY ON CLOCK B'S INTR.
LOOP:                                           ;GET NEXT RATE.

                                                ;IF NOT END THEN CONTINUE.
                                                ;CLEAR THE OFFSET
                                                ;LOOK AT FIRST RATE
1$:                                             ;IS THIS RATE INHIBITED?

                                                ;CLR FLAG INDICATING CLOCK A HAS INTERRUPTED.
                                                ;CLR FLAG INDICATED CLOCK B HAS INTERRUPTED.
                                                ;PICK UP OFFSET
                                                ;GO SET THE RATE + START THE CLOCK.

                                                ;
                                                ;THE FOLLOWING (LISTP) ARE POINTERS TO VARIOUS RATE
                                                ;ROUTINES. THEY ARE USED BY "LOOP". "LOOP"  GENERATES
                                                ;AN OFFSET OF A RATE WE WISH TO EXERCISE. THE OFFSET
                                                ;IS STORED IN R1. WE INDEX "LISTP" BY R1 (JMP @LISTP(R1))
                                                ;TO GET THE ADDRESS OF THE RATE ROUTINE TO EXERCISE.
```

```
                                        ;
                                        ;
002136' 000001          LISTP:  .WORD   1               ;
002140' 002240'                 .WORD   RATE0           ;POINTER TO 1MHZ ROUTINE
002142' 002304'                 .WORD   RATE1           ;POINTER TO 100KHZ ROUTINE
002144' 002350'                 .WORD   RATE2           ;POINTER TO 10KHZ ROUTINE
002146' 002414'                 .WORD   RATE3           ;POINTER TO 1KHZ ROUTINE
002150' 002460'                 .WORD   RATE4           ;POINTER TO 100HZ ROUTINE
002152' 002524'                 .WORD   RATE5           ;POINTER TO RANDOM ROUTINE
002154' 002654'                 .WORD   RATE6           ;POINTER TO LINE FREQ. ROUTINE
002156' 002720'                 .WORD   RATE7           ;POINTER TO FEED B TO A ROUTINE
                                        ;
                                        ;
                                        ;THE FOLLOWING (RATEAL) ARE THE PRESET VALUES THAT THE
                                        ;VARIOUS RATE ROUTINES NEED. THEY ARE LOADED INTO
                                        ;CLOCK A'S PRESET BUFFER. "RATEAL" IS INDEXED BY
                                        ;AN OFFSET IN R1 BY THE RATE ROUTINES TO GET THE
                                        ;PRESET VALUE
                                        ;
002160' 000001          RATEAL: .WORD   1               ;OFFSET ZERO,NO RATE.
002162' 036260                  .WORD   -50000.         ;VALUE FOR 1MHZ PRESET.
002164' 036260                  .WORD   -50000.         ;PRESET VALUE FOR 100 KHZ
002166' 154360                  .WORD   -10000.         ;PRESET VALUE FOR 10 KHZ
002170' 176030                  .WORD   -1000.          ;PRESET VALUE FOR 1 KHZ
002172' 177634                  .WORD   -100.           ;PRESET VALUE FOR 100 HZ
002174' 000000                  .WORD   0               ;PRESET VALUE FOR RANDOM
002176' 177704                  .WORD   -60.            ;PRESET VALUE FOR LINE FREQ.
002200' 170272                  .WORD   -3910.          ;PRESET VALUE FOR FEED B TO A
                                        ;
                                        ;
                                        ;THE FOLLOWING (RATEBL) ARE THE PRESET VALUES THAT THE
                                        ;VARIOUS RATE ROUTINES NEED. THEY ARE LOADED INTO CLOCK B'S
                                        ;PRESET BUFFER. "RATEBL" IS INDEXED BY R1 BY THE RATE
                                        ;ROUTINES TO GET THE PRESET VALUES.
                                        ;
002202' 000001          RATEBL: .WORD   1               ;OFFSET ZERO,NO RATE.
002204' 000000                  .WORD   0               ;PRESET VALUE FOR 1 MHZ.
002206' 000000                  .WORD   0               ;PRESET VALUE FOR 100 KHZ.
002210' 000000                  .WORD   0               ;PRESET VALUE FOR 10 KHZ.
002212' 000000                  .WORD   0               ;PRESET VALUE FOR 1 KHZ.
002214' 177660                  .WORD   -80.            ;PRESET VALUE FOR 100 HZ.
002216' 000000                  .WORD   0               ;PRESET VALUE FOR RANDOM
002220' 177730                  .WORD   -40.            ;PRESET VALUE FOR LINE FREQ.
002222' 000000                  .WORD   0               ;PRESET VALUE FOR FEED B TO A.
                                        ;
                                        ;
                                        ;THE FOLLOWING (RSAL) IS USED BY THE RANDOM
                                        ;RATE ROUTINE (RATE5). THEY ARE THE VALUES NEEDED
                                        ;TO BE PUT INTO THE CLOCK'S CSR FOR A PARTICULAR RATE.
                                        ;
002224' 000000          RSAL:   .WORD   0               ;OFFSET ZERO,NO RATE.
002226' 000503                  .WORD   503             ;1 MHZ, GO.
002230' 000505                  .WORD   505             ;100 KHZ, GO.
```



```
002232' 000107                  .WORD   107             ;10 KHZ, GO.
002234' 000111                  .WORD   111             ;1 KHZ, GO.
002236' 000113                  .WORD   113             ;100 HZ, GO.
                                        ;*
                                        ;*THIS ROUTINE PRESETS CLOCK A AND B FOR
                                        ;*1 MHZ RATE     CLOCK A INTRS IN 1/20 SEC., 25 TIMES.
                                        ;*               CLOCK B INTRS IN 0.256 MILLI. SEC.
                                        ;*
002240'                 RATE0:
002240' 005077  175760          CLR     @ASR            ;CLEAR CLOCK A.
002244' 005077  175762          CLR     @BSR            ;CLEAR CLOCK B.
002250' 016177  002160' 175750  MOV     RATEAL(R1),@ABR ;PRESET COUNT IN CLOCK A.
002256' 012777  000503  175740  MOV     #503,@ASR       ;START CLOCK A.
002264' 016177  002202' 175742  MOV     RATEBL(R1),@BBR ;PRESET COUNT IN CLOCK B.
002272' 012777  000103  175732  MOV     #103,@BSR       ;START CLOCK B.
002300' 104400  000000'                                 ;NOW WAIT FOR INTERRUPT.
                                EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
                                        ;*
                                        ;*THIS ROUTINE PRESETS CLOCK A AND B FOR
                                        ;*100 KHZ RATE   CLOCK A INTRS IN .5 SEC., TWICE.
                                        ;*               CLOCK B INTRS IN 2.56 MILLI SEC.
                                        ;*
002304'                 RATE1:
002304' 005077  175714          CLR     @ASR            ;CLEAR CLOCK A.
002310' 005077  175716          CLR     @BSR            ;CLEAR CLOCK B.
002314' 016177  002160' 175704  MOV     RATEAL(R1),@ABR ;PRESET COUNT IN CLOCK A.
002322' 012777  000505  175674  MOV     #505,@ASR       ;START CLOCK A.
002330' 016177  002202' 175676  MOV     RATEBL(R1),@BBR ;PRESET COUNT IN CLOCK B.
002336' 012777  000105  175666  MOV     #105,@BSR       ;START CLOCK B.
002344' 104400  000000'                                 ;NOW WAIT FOR INTERRUPT.
                                EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
                                        ;*
                                        ;*THIS ROUTINE PRESETS CLOCK A AND B FOR
                                        ;*10 KHZ RATE    CLOCK A INTRS IN 1.0 SEC.
                                        ;*               CLOCK B INTRS IN 25.6 MILLI SEC.
                                        ;*
002350'                 RATE2:
002350' 005077  175650          CLR     @ASR            ;CLEAR CLOCK A.
002354' 005077  175652          CLR     @BSR            ;CLEAR CLOCK B.
002360' 016177  002160' 175640  MOV     RATEAL(R1),@ABR ;PRESET COUNT IN CLOCK A.
002366' 012777  000107  175630  MOV     #107,@ASR       ;START CLOCK A.
002374' 016177  002202' 175632  MOV     RATEBL(R1),@BBR ;PRESET COUNT IN CLOCK B.
002402' 012777  000107  175622  MOV     #107,@BSR       ;START CLOCK B.
002410' 104400  000000'                                 ;NOW WAIT FOR INTERRUPT.
                                EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
                                        ;*
                                        ;*THIS ROUTINE PRESETS CLOCK A AND B FOR
                                        ;*1 KHZ RATE     CLOCK A INTRS IN 1.0 SEC.
                                        ;*               CLOCK B INTRS IN 0.256 SEC.
```

```
                                        ;*
                                        ;*
002414'                         RATE3:
002414' 005077  175604          CLR     @ASR            ;CLEAR CLOCK A.
002420' 005077  175606          CLR     @BSR            ;CLEAR CLOCK B.
002424' 016177  002160' 175574  MOV     RATEAL(R1),@ABR ;PRESET COUNT IN CLOCK A.
002432' 012777  000111  175564  MOV     #111,@ASR       ;START CLOCK A.
002440' 016177  002202' 175566  MOV     RATEBL(R1),@BBR ;PRESET COUNT IN CLOCK B.
002446' 012777  000111  175556  MOV     #111,@BSR       ;START CLOCK B.
                                                        ;NOW WAIT FOR INTERRUPT.
002454' 104400  000000'         EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
                                        ;*
                                        ;*THIS ROUTINE PRESETS CLOCK A AND B FOR
                                        ;*100 HZ RATE   CLOCK A INTRS IN 1.0 SEC.
                                        ;*              CLOCK B INTRS IN 0.8 SEC.
                                        ;*
002460'                         RATE4:
002460' 005077  175540          CLR     @ASR            ;CLEAR CLOCK A.
002464' 005077  175542          CLR     @BSR            ;CLEAR CLOCK B.
002470' 016177  002160' 175530  MOV     RATEAL(R1),@ABR ;PRESET COUNT IN CLOCK A.
002476' 012777  000113  175520  MOV     #113,@ASR       ;START CLOCK A.
002504' 016177  002202' 175522  MOV     RATEBL(R1),@BBR ;PRESET COUNT IN CLOCK B.
002512' 012777  000113  175512  MOV     #113,@BSR       ;START CLOCK B.
                                                        ;NOW WAIT FOR INTERRUPT.
002520' 104400  000000'         EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

                                        ;*THIS ROUTINE PRESETS CLOCK A + B FOR
                                        ;*RANDOM RATES
002524' 004767  000436          RATE5:  JSR     PC,RANDOM       ;GET 2 RANDOM NUMBERS.

002530' 042767  177771  175516          BIC     #177771,RANA    ;MAKE NUMBER < 10.
002536' 042767  177771  175512          BIC     #177771,RANB    ;MAKE 2ND NUMBER < 10.

                                        ;NUMBERS MUST BE 2, 4, OR 6
002544' 005767  175504          3$:     TST     RANA            ;IS NUMBER ZERO?
002550' 001003                  BNE     4$              ;NO-GO AHEAD.
002552' 062767  000002  175474  ADD     #2,RANA         ;MAKE IT NON-ZERO

002560' 005767  175472          4$:     TST     RANB            ;IS NUMBER      ZERO?
002564' 001003                  BNE     5$              ;NO GO AHEAD.
002566' 062767  000002  175462  ADD     #2,RANB         ;MAKE IT NON-ZERO.
002574'                         5$:

002574' 005077  175424          CLR     @ASR            ;CLEAR CLOCK A
002600' 005077  175426          CLR     @BSR            ;CLEAR CLOCK B
002604' 016701  175444          MOV     RANA,R1
002610' 010167  175436          MOV     R1,OFF          ;RECORD THE OFFSET.
002614' 016177  002160' 175404  MOV     RATEAL(R1),@ABR ;PRESET CLOCK A.
002622' 016177  002224' 175374  MOV     RSAL(R1),@ASR   ;START CLOCK A.
002630' 016701  175422          MOV     RANB,R1
002634' 016177  002202' 175372  MOV     RATEBL(R1),@BBR ;PRESET CLOCK B
002642' 016177  002224' 175362  MOV     RSAL(R1),@BSR   ;START CLOCK B
```



```
002650' 104400  000000'         EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
                                        ;*
                                        ;*THIS ROUTINE PRESETS CLOCK A AND B FOR
                                        ;*LINE FREQ RATE        CLOCK A INTRS. IN 1.0 SEC
                                        ;*                      CLOCK B INTRS. IN 0.8 SEC
                                        ;*
002654' 005077  175344          RATE6:  CLR     @ASR            ;CLEAR CLOCK A.
002660' 005077  175346          CLR     @BSR            ;CLEAR CLOCK B.
002664' 016177  002160' 175334  MOV     RATEAL(R1),@ABR ;PRESET COUNT IN A.
002672' 012777  000117  175324  MOV     #117,@ASR       ;START CLOCK A.
002700' 016177  002202' 175326  MOV     RATEBL(R1),@BBR ;PRESET COUNT IN B.
002706' 012777  000117  175316  MOV     #117,@BSR       ;START CLOCK B.

002714' 104400  000000'         EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERPUPT.
                                        ;*
                                        ;*THIS ROUTINE PRESETS CLOCK A + B FOR
                                        ;*FEED B TO A RATE
                                        ;*CLOCK A INTERRUPTS IN 1.0 SECS.
                                        ;*
002720' 005077  175300          RATE7:  CLR     @ASR            ;CLEAR CLOCK A.
002724' 005077  175302          CLR     @BSR            ;CLEAR CLOCK B.
002730' 005267  175326          INC     BIFLG
002734' 016177  002160' 175264  MOV     RATEAL(R1),@ABR ;PRESET CLOCK A.
002742' 012777  000101  175254  MOV     #101,@ASR       ;START CLOCK A.
002750' 016177  002202' 175256  MOV     RATEBL(R1),@BBR ;PRESET CLOCK B.
002756' 012777  000043  175246  MOV     #43,@BSR        ;START CLOCK B.

002764' 104400  000000'         EXIT$,BEGIN             ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

                                        ;*
                                        ;*INTERRUPT SERVICE ROUTINE
                                        ;*FOR CLOCK A.
                                        ;*
002770' 005267  175264          INSERV: INC     AIFLG           ;INDICATE CLOCK A HAS INTERRUPTED

002774'                         2$:
002774' 026727  175252  000002          CMP     OFF,#2          ;ARE WE RUNNING 1MHZ RATE?
003002' 001005                  BNE     3$              ;IF NOT ( 3$
003004' 026727  175250  000024          CMP     AIFLG,#20.      ;1 MHZ, 1 SEC. UP?
003012' 001412                  BEQ     4$              ;YES - STOP
003014' 000002                  RTI                     ;NO, ALLOW ANOTHER COUNT.
003016' 026727  175230  000004  3$:     CMP     OFF,#4          ;100 KHZ RATE?
003024' 001005                  BNE     4$              ;NO - 4$
003026' 026727  175226  000002          CMP     AIFLG,#2        ;YES, COUNTED TWICE?
003034' 001401                  BEQ     4$              ;YES - 4$.
003036' 000002                  RTI                     ;NO - COUNT ONE MORE TIME.
003040'                         4$:
003040' 005767  175216          TST     BIFLG           ;HAS CLOCK B INTERRUPTED?
003044' 001030                  BNE     5$              ;YES THEN 5$
003046' 017767  175152  175030  MOV     @ASR,ASTAT      ;RECORD CONTENTS OF A'S CSR.
003054' 017767  175152  175020  MOV     @BSR,ACSR       ;RECORD CONTENTS OF B'S CSR.
```

```
818  003062' 016767  175144  175010          MOV     BSR,CSRA        ;RECORD B'S ADDR.
819  003070' 005077  175130                  CLR     @ASR            ;STOP CLOCK A.
820  003074' 005077  175132                  CLR     @BSR            ;STOP CLOCK B.
821                                          ;----------------------------------------------------------
822  003100' 000004  000000' 003106'         PIRQ$,BEGIN,1$          ; QUEUE UP TO CONTINUE AT 1$ AND RTI
823                                          ;----------------------------------------------------------
824
825  003106' 012767  000011  174772  1$:     MOV     #11,ERRTYP      ;NO INTERRUPT
826                                          ;**********************************************************
827  003114' 104405  000000' 000000          HRDER$,BEGIN,NULL       ;ERROR CLOCK B FAILED TO INTERRUPT.
828                                          ;**********************************************************
829
830  003122' 000167  176732                  JMP     LOOP
831  003126'                         5$:
832                                          ;----------------------------------------------------------
833  003126' 000004  000000' 003134'         PIRQ$,BEGIN,6$          ; QUEUE UP TO CONTINUE AT 6$ AND RTI
834                                          ;----------------------------------------------------------
835  003134' 005077  175064         6$:      CLR     @ASR            ;STOP CLOCK A.
836  003140' 005077  175066                  CLR     @BSR            ;STOP CLOCK B.
837  003144' 104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
838                                                                  ;MONITOR SHALL TEST END OF PASS
839  003150' 000167  176704                  JMP     LOOP
840
841                                                  ;*
842                                                  ;*INTERRUPT SERVICE ROUTINE
843                                                  ;*FOR CLOCK B.
844                                                  ;*
845
846  003154' 005267  175102         INSERB:  INC     BIFLG           ;INDICATE CLOCK B HAS INTERRUPTED.
847  003160' 005077  175046                  CLR     @BSR            ;STOP CLOCK B
848  003164' 000002                          RTI                     ;EXIT
849
850                                                  ;*
851                                                  ;*THIS SUBROUTINE GENERATES
852                                                  ;*TWO RANDOM NUMBERS
853                                                  ;*
854                                                  ;*CALL = JSR PC, RANDOM
855                                                  ;*
856                                                  ;*RETURNS WITH NUMBERS IN RANDA AND RANDB
857                                                  ;*
858
859  003166' 066767  175062  175062  RANDOM: ADD     RANA,RANB
860  003174' 005567  175054                  ADC     RANA
861  003200' 066767  175052  175046          ADD     RANB,RANA
862  003206' 005567  175044                  ADC     RANB
863  003212' 000207                          RTS     PC
864
865
866
867          000001                          .END
```



```
ABR        000226R     262#  291*  450*  451   457   463*  464   469   481*  482   488   494*  495
                       500   663*  679*  695*  711*  727*  756*  772*  788*
ACR        000230R     263#  293*
ACSR       000102R     240#  327*  328   340*  358*  359   371*  389*  390   402*  420*  421   433*
                       451*  452   464*  482*  483   495*  513*  514   526*  544*  545   557*  817*
ADDR       000006R     206#  286
ADDR22=    001000      258#
AIFLG      000260R     280#  583*  800*  805   810
ASB        000106R     244#
ASR        000224R     261#  286*  289   315   326*  327   333   339*  340   345   357*  358   364
                       370*  371   376   661*  664*  677*  680*  693*  696*  709*  712*  725*  728*
                       752*  757*  770*  773*  785*  789*  816   819*  835*
ASTAT      000104R     242#  325*  326   328   344*  356*  357   359   375*  387*  388   390   406*
                       418*  419   421   437*  449*  450   452   468*  480*  481   483   499*  511*
                       512   514   530*  542*  543   545   561*  816*
AVECT      000240R     269#  287*  301   304   570*
AVECT2     000242R     270#  304*  305*  572*
AWAS       000110R     245#
BBR        000234R     266#  297*  512*  513   519   525*  526   531   543*  544   550   556*  557
                       562   665*  681*  697*  713*  729*  759*  774*  790*
BCR        000236R     267#  299*
BEGIN      000000R     203#  330   331   336   342   343   348   361   362   367   373   374   379
                       392   393   398   404   405   410   423   424   429   435   436   441   454
                       455   460   466   467   472   485   486   491   497   498   503   516   517
                       522   528   529   534   547   548   553   559   560   565   668   684   700
                       716   732   762   777   793   822   827   833   837
BIFLG      000262R     281#  584*  787*  814   846*
BIT0     = 000001      258#
BIT1     = 000002      258#
BIT10    = 002000      258#
BIT11    = 004000      258#
BIT12    = 010000      258#
BIT13    = 020000      258#
BIT14    = 040000      258#
BIT15    = 100000      258#
BIT2     = 000004      258#
BIT3     = 000010      258#
BIT4     = 000020      258#
BIT5     = 000040      258#
BIT6     = 000100      258#
BIT7     = 000200      258#  569
BIT8     = 000400      258#
BIT9     = 001000      258#
BREAK$=    104407      258#  330   331   342   343   361   362   373   374   392   393   404   405
                       423   424   435   436   454   455   466   467   485   486   497   498   516
                       517   528   529   547   548   559   560
                       572   573
BR1        000012R     208#
BR2        000013R     209#
BSR        000232R     265#  295*  388*  389   395   401*  402   407   419*  420   426   432*  433
                       438   662*  666*  678*  682*  694*  698*  710*  714*  726*  730*  753*  760*
                       771*  775*  786*  791*  817   818   820*  836*  847*
BTOD$    = 104421      258#
BVECT      000244R     272#  303*  306   571*
BVECT2     000246R     273#  306*  307*  573*
CDATA$=    104412      258#
CONFIG     000056R     228#
```

```
CSRA    000100R          238#  333*  345*  364*  376*  395*  407*  426*  438*  457*  469*  488*  500*
                         519#  531*  550*  562*  818*
DATCK$= 104411           258#
DATER$= 104404           258#
DVID1   000014R          210#
EMDIT$= 104413           258#  837
END$    = 104410         258#
ERRTYP  000106R          243#  284*  334*  346*  365*  377*  396*  408*  427*  439*  458*  470*  489*
                         501#  520*  532*  551*  563*  825*
EXIT$ = 104400           258#  668   684   700   716   732   762   777   793
GETPA$= 104415           258#
GWBUF$= 104414           258#
HRDCNT  000044R          223#
HRDER$= 104405           258#  336   348   367   379   398   410   429   441   460   472   491   503
                         522   534   553   565   827
HRDPAS  000050R          225#
ICONT   000036R          220#
ICOUNT  000040R          221#
IDNUM   000122R          250#
INIT    000030R          217#
INSERB  003154R          571   846#
INSERV  002770R          570   800#
INTR    000120R          249#
LISTP   002136R          586   596#
LOG1    000430R          315#
LOG2    000434R          325#
LOG3    000572R          341   356#
LOG4    000730R          372   387#
LOG5    001066R          403   418#
LOG6    001224R          434   449#
LOG7    001362R          465   480#
LOG8    001520R          496   511#
LOG9    001656R          527   542#
LOOP    002060R          574#  581   830   839
MAP22$= 104416           258#
MODNAM  000000R          204#
MODSP   000224R          218   256#
MSGN$ = 104403           258#
MSGS$ = 104402           258#
MSG$  = 104401           258#
NULL  = 000000           258#  336   348   367   379   398   410   429   441   460   472   491   503
                         522   534   553   565   827
OFF     000252R          277#  577*  578*  585   755*  803   808
OPEN  = 000000           205   211   212   213   214   231   232   233   234   235   236   237   238
                         240   242   244   245   247   248   249   258#
OTQA$ = 104420           258#
PASCNT  000034R          219#
PIRQS = 000004           258#  822   833
POPSP = 005726           258#
POPSP2= 022626           258#
PRTY  = 000000           258#
PRTY0 = 000000           258#
PRTY1 = 000040           258#
PRTY2 = 000100           258#
PRTY3 = 000140           258#
PRTY4 = 000200           258#
```



```
PRTY5 = 000240           258#
PRTY6 = 000300           258#  209   258#
PRTY7 = 000340           258#
PS    = 177776           258#
PSW   = 177776           258#
PUSH  = 005746           258#
PUSH2 = 024646           258#
RAMA    000254R          278#  568*  739*  743   745*  754   859   860*  861*
RAMB    000256R          279#  740*  747   749*  758   859*  861   862*
RANDOM  003166R          737   859#
RANDS = 104417           258#
RANNUM  000054R          227#
RATEAL  002160R          614#  663   679   695   711   727   756   772   788
RATEBL  002202R          631#  665   681   697   713   729   759   774   790
RATEP   000250R          276#  569*  574*  578*  580
RATE0   002240R          597   660#
RATE1   002304R          598   676#
RATE2   002350R          599   692#
RATE3   002414R          600   708#
RATE4   002460R          601   724#
RATE5   002524R          602   737#
RATE6   002554R          603   770#
RATE7   002720R          604   785#
RESTRT  002014R          246   558   568#
RES1    000056R          229#
RES2    000060R          230#
RSAL    002224R          647#  757   760
RSTRT   000112R          246#
SBADR   000102R          239#
SOFCNT  000042R          222#
SOFER$= 104406           258#
SOFPAS  000046R          224#
SPOINT  000032R          218#
SPSIZ = 000040             1#  251
SR1     000016R          211#  580
SR2     000020R          212#
SR3     000022R          213#
SR4     000024R          214#
START   000266R          217   284#
STAT    000026R          216#
SVR0    000062R          231#
SVR1    000064R          232#
SVR2    000066R          233#
SVR3    000070R          234#
SVR4    000072R          235#
SVR5    000074R          236#
SVR6    000076R          237#
SYSCNT  000052R          226#
TRPDFD= 000022           258#
TRY     000264R          282#
VECTOR  000010R          207#  287
WASADR  000104R          241#
WDFR    000116R          248#
WDTO    000114R          247#
XFLAG   000005R          205#
```

```
. ABS.  000000      000
        003214      001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 XKWDB0,XKWDB0/SOL/CRF:SYM=DDXCOM,XKWDB0
 RUN-TIME: 1 2 .4 SECONDS
 RUN-TIME RATIO: 45/5=8.8
 CORE USED:  7K  (13 PAGES)
```