```
.REM _
```

IDENTIFICATION

PRODUCT CODE: AC-E688G-MC

PRODUCT NAME: CXKWAG0 DEC/X11 KW11-L LINE CLOCK MODULE

DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

1. ABSTRACT

THE KWA IS AN IOMOD THAT EXERCISES THE KW11-L LINE FREQUENCY CLOCK.
IT IS ALSO USED AS THE SYSTEM CLOCK BY THE MONITOR.

2. REQUIREMENTS:

HARDWARE: ONE KW11-L OPTION MODULE
SOFTWARE: CAN BE USED AS A SYSTEM CLOCK

STORAGE:: KWA REQUIRES:
1. DECIMAL WORDS: 467
2. OCTAL WORDS: 0723
3. OCTAL BYTES: 1646

3. PASS DEFINITION

ONE PASS CONSISTS OF 3600. INTERUPTS AT 60HZ OR 3000. AT 50HZ

4. EXECUTION TIME

THE KWA RUNNING ALONE ON A PDP11/05 PROCESSOR TAKE APPROXIMATELY 1 MINUTE

5. CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:

DEVADR: 177564, VECTOR: 100, BR1: 6, DEVCNT: 1, SR1:0

REQUIRED PARAMETERS:

NONE

6. DEVICE/OPTION SETUP:

NONE

7. MODULE OPERATION:

TEST SEQUENCE:

A. SET UP DEVICE ADDRESS AND VECTOR, ANY TIME MESSAGES, AND 50 OR 60 HZ.
B. ENABLE INTERUPTS
C. COUNT THE INTERUPT, 50/60 INTERUPTS = 1 SECOND
D. REPEAT B THRU C 3600. TIMES (60HZ) OR 3000. TIMES (50HZ)
E. REPORT ENDPAS BUT KEEP CLOCK RUNNING
F. AT RESTART, CHECK IF IT'S TIME FOR MESSAGE
G. CONTINUE AT C

SINCE THE KW11L IS A GO/NO GO TYPE OF OPTION THERE ARE NO ERROR PRINTOUTS. FAILURE TO GET AN END OF PASS REPORT INDICATES NO GO.

IF THE CLOCK HAS NOT BEEN RESTARTED FROM AN ENDPAS CALL AFTER 15. MINUTES HAVE ELAPSED, IT WILL MAKE A JMP INTO THE MONITOR (CKHUNG) TO CHECK FOR ANY HUNG MODULES.

8. OPERATION OPTIONS:

VALID SR1 VALUES:

SR1	HERTZ	TIME MSG EVERY X MINUTES	
0	60	NEVER	
1	50	NEVER	
2	60	5	
3	50	5	
4	60	15	; ALL TIMES ARE APPROXIMATE
5	50	15	
6	60	60	
7	50	60	

9. NON-STANDARD PRINTOUTS

NONE: ALL PRINTOUTS HAVE THE STANDARD FORMATS DESCRIBED IN THE DEC/X11 DOCUMENT.

```
                ;KW11L DEC/X11 EXERCISER MODULE

000000'                 IOMOD   <KWAG >,177546,100,6,,,50.,,11
000000'                 MODULE  140000,KWAG ,177546,100,6,,,50.,,11
                        .TITLE  KWAG DEC/X11 SYSTEM EXERCISER MODULE
                ;       DDXCOM  VERSION  6      23-MAY-78
                                .LIST   BIN
                                ;*****************************************************************************
000000'                         ;BEGIN:
000000' 053513  043501     040  MODNAM: .ASCII  /KWAG / ;MODULE NAME.
000005'     000                 XFLAG:  .BYTE   0PEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 177546                  ADDR:   177546+0                ;1ST DEVICE ADDR.
000010' 000100                  VECTOR: 100+0                   ;1ST DEVICE VECTOR.
000012'     300                 BR1:    .BYTE   PRTY6+0         ;1ST BR LEVEL.
000013'     000                 BR2:    .BYTE   PRTY+0          ;2ND BR LEVEL.
000014' 000001                  DVID1:  +1                      ;DEVICE INDICATOR 1.
000016' 000000                  SR1:    OPEN                    ;SWITCH REGISTER 1
000020' 000000                  SR2:    OPEN                    ;SWITCH REGISTER 2
000022' 000000                  SR3:    OPEN                    ;SWITCH REGISTER 3
000024' 000000                  SR4:    OPEN                    ;SWITCH REGISTER 4
                                ;*****************************************************************************
000026' 140000                  STAT:   140000                  ;STATUS WORD.
000030' 000224'                 INIT:   START                   ;MODULE START ADDR.
000032' 000224'                 SPOINT: MODSP                   ;MODULE STACK POINTER.
000034' 000000                  PASCNT: 0                       ;PASS COUNTER.
000036' 000062                  ICONT:  50.                     ;# OF ITERATIONS PER PASS=50.
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' 000706'                 RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114' 000000                  WDTO1:  OPEN                    ;WORDS TO MEMORY PER ITERATION
000116' 000000                  WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
000120' 000000                  INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
```



```
000122' 000011                  IDNUM:  11                      ;MODULE IDENTIFICATION NUMBER=11
        000040                          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224'                         MODSP:
                                ;*****************************************************************************
```

```
                                        .GLOBL  KW11L,HMS,CLOCK,CKHUNG,LCLEAR
000224' 012767  000062  177604  START:  MOV     #50.,ICONT      ;NEEDED IN CASE OF RESTART
000232' 005767  000000G                 TST     CLOCK           ;ANY TIMING ANALYSIS ERROR?
000236' 001002                          BNE     1$              ;BR IF NOT
000240' 104410  000000'                 ENDS,BEGIN              ; CLOCK NOT AVAILABLE, DROP IT
000244' 005767  177564          1$:     TST     PASCNT          ; POWER FAIL RESTART ?
000250' 001002                          BNE     2$              ; YES, DON'T ZERO TIME INFO
000252' 004767  000604                  JSR     PC,LCLEAR       ;GO CLEAR PASS TIME TABLE
000256' 012767  000001  000644  2$:     MOV     #1,RUNING       ;SHOW WE ARE IN A PASS
000264' 016701  177526                  MOV     SR1,R1          ;WHAT IS MSG INTERVAL?
000270' 006201                          ASR     R1              ;GET RID OF 50/60 HZ BIT
000272' 012767  000454  000620          MOV     #300.,INTER     ;SETUP FOR 5 MIN. MESSAGES
000300' 022701  000001                  CMP     #1,R1           ; DOES HE WANT 5 MIN. MESSAGES ?
000304' 001411                          BEQ     3$              ;BR IF YES
000306' 062767  001130  000604          ADD     #600.,INTER     ;NO ADD 10 MORE MINUTES
000314' 022701  000002                  CMP     #2,R1           ;IS IT 15 MINUTES?
000320' 001403                          BEQ     3$              ;BR IF YES
000322' 062767  005214  000570          ADD     #2700.,INTER    ;NO, MUST BE 60 MINUTES
000330' 016767  000564  000560  3$:     MOV     INTER,MSGTIM    ;SO ADD 45 MORE MINUTES
000336' 005067  000570                  CLR     MTIME           ;MTIME IS COUNT FOR WHEN TO TYPE MSG
000342' 016701  177440                  MOV     ADDR,R1         ; GET DEVICE ADDRESS
000346' 016767  177434  000550          MOV     ADDR,CLKCSR     ;AND PUT IT IN CLKCSR
000354' 016700  177430                  MOV     VECTOR,R0       ;GET VECTOR
000360' 012720  000456'                 MOV     #KLKINT,(R0)+   ;AND STORE ITS ADDRESS THERE
000364' 177422                          MOV     #BR7,(R0)+      ;SET UP THE BR LEVEL
000370' 012767  000062  000524          MOV     #50.,BASE       ;ASSUME ITS A 50HZ MACHINE
000376' 032767  000001  177412          BIT     #1,SR1          ;IS IT REALLY?
000404' 001006                          BNE     4$              ;BR AROUND IF YES, ELSE
000406' 062767  000012  000506          ADD     #10.,BASE       ;ADD 10 MORE TO = 60HZ
000414' 062767  000012  177414          ADD     #10.,ICONT      ;FOR 60 CPS,ICONT=60
000422' 016767  000474  000504  4$:     MOV     BASE,TIMCTR     ;COUNT TICS TO GET SECONDS
000430' 016767  177402  177462          MOV     ICONT,INTR      ;ICONT WILL BE # OF INTERRUPTS

000436' 016767  000460  000462  CLKGO:  MOV     BASE,CLKCTR     ;COUNT TICS TO GET SECS FOR ENDPASS
000444' 012777  000100  000452          MOV     #BIT6,@CLKCSR   ;START THE CLOCK GOING
000452' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

                                        ;INTERUPT SERVICE ROUTINE , EVEN WHEN MODULE IS BETWEEN PASSES

000456' 005077  000442          KLKINT: CLR     @CLKCSR         ; DISABLE INTERUPTS
000462' 005367  000446                  DEC     TIMCTR          ;KEEP TRACK OF TICS FOR MSG
000466' 001033                          BNE     1$              ;BR IF NOT 1 SECOND ELAPESED
000470' 062767  000001  000512          ADD     #1,TIME         ;COUNT A SECOND IN TOTAL TIME
000476' 005567  000510                  ADC     TIMEXT          ; EXTENDED BITS FOR THE TIME WORD
000502' 005267  000424                  INC     MTIME           ;COUNT A SECOND FOR MSG TIME
000506' 016767  000410  000420          MOV     BASE,TIMCTR     ;COUNT TICS AGAIN FOR NEXT SECOND
000514' 005767  000410                  TST     RUNING          ; ARE WE BETWEEN PASSES ?
000520' 001016                          BNE     1$              ; NO, CONTINUE
000522' 016767  000462  000406          MOV     TIME,TMPTIM     ; GET THE PRESENT TIME
000530' 166767  000404  000400          SUB     HLDTIM,TMPTIM   ; GET ELAPSED TIME SINCE LAST EOP
000536' 022767  001604  000372          CMP     #900.,TMPTIM    ; HAS 15. MINUTE LIMIT BEEN PASSED ?
000544' 101004                          BHI     1$              ; NO, CONTINUE
000546' 062767  001604  000364          ADD     #900.,HLDTIM    ; YES, CHECK AGAIN AFTER 15. MINUTES
000554' 000412                          BR      3$              ; CHECK FOR ANY "HUNG" MODULES
```



```
000556' 005767  000346          1$:     TST     RUNING          ;ARE WE BETWEEN PASSES?
000562' 001403                          BEQ     2$              ;BR IF YES
000564' 005367  000336                  DEC     CLKCTR          ;ELSE COUNT ANOTHER TIC
000570' 001422                          BEQ     5$              ;IF COUNTED 60, GO ADD A SEC
000572' 012777  000100  000324  2$:     MOV     #BIT6,@CLKCSR   ;START CLOCK OFF AGAIN
000600' 000002                          RTI

000602'                         3$:
                                        ;-------------------------------------------------------------
000602' 000004  000000' 000610'         PIRQ$,BEGIN,4$          ; QUEUE UP TO CONTINUE AT 4$ AND RTI
                                        ;-------------------------------------------------------------
000610' 012777  000100  000306  4$:     MOV     #BIT6,@CLKCSR   ; TURN THE CLOCK BACK ON
000616' 010546                          MOV     R5,-(SP)        ;SAVE R5
000620' 012705  001644'                 MOV     #R5STK,R5       ;SET UP R5 STACK
000624' 004767  000000G                 JSR     PC,CKHUNG       ; BACK TO MONITOR. CHECK FOR HUNG MODULES
000630' 012605                          MOV     (SP)+,R5        ;RESTORE R5
000632' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

000636'                         5$:
                                        ;-------------------------------------------------------------
000636' 000004  000000' 000644'         PIRQ$,BEGIN,6$          ; QUEUE UP TO CONTINUE AT 6$ AND RTI
                                        ;-------------------------------------------------------------

000644'                         6$:
000644' 016700  000252                  MOV     BASE,R0         ;GET BASE INTO R0
000650' 005300                          DEC     R0              ;SET UP R0
000652' 026700  177162                  CMP     ICOUNT,R0       ;IS IT TIME FOR END OF PASS??
000656' 001005                          BNE     7$              ;BRANCH IF NOT
000660' 005067  000244                  CLR     RUNING          ;CLEAR BETWEEN PASS FLAG
000664' 016767  000320  000246          MOV     TIME,HLDTIME    ;SAVE THE TIME OF EOP
000672' 012777  000100  000224  7$:     MOV     #BIT6,@CLKCSR   ;KEEP CLOCK GOING
000700' 104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                ;MONITOR SHALL TEST END OF PASS
000704' 000654                          BR      CLKGO           ; NO, KEEP ON TICKIN'
000706'                         RESTRT:

000706' 005767  177122                  TST     PASCNT          ; THIS IS FOR CSS BUS SWITCHES
000712' 001002                          BNE     REST            ; CONTINUE
000714' 000167  177304                  JMP     START           ; BEGIN AT START
000720' 026767  000206  000172  REST:   CMP     MTIME,INTER     ;IS IT TIME FOR A MSG?
000726' 103450                          BLO     1$              ;NO, BRANCH
000730' 005067  000176                  CLR     MTIME           ;YES, RESET MSG TIME COUNT
000734' 026727  177056  000002          CMP     SR1,#2          ;ARE ANY TIME MSG'S WANTED
000742' 002442                          BLT     1$              ;BR IF NOT

                                        ;+
                                        ; SET UP R5 STACK AND CONVERT TIME AND TIMEXT TO HOURS, MINUTES AND SECONDS.
                                        ;       CALLING SEQUENCE: CALL HMS IN <TIME,TIMEXT> OUT <R0>
                                        ;-

000744' 010046                          MOV     R0,-(SP)        ;SAVE R0
000746' 010546                          MOV     R5,-(SP)        ;SAVE R5
000750' 012705  001644'                 MOV     #R5STK,R5       ;SET UP R5 STACK
000754' 162705  000002                  SUB     #1*#2,R5        ;SAVE SPACE ON STACK FOR OUTPUT
```

```
                                                        ;ARGUMENT FOR HMS CALL
001062' 010546                          MOV     R5,-(SP)        ;SAVE R5 STACK POINTER ON R6 STACK
000762' 016745  000224                  MOV     TIMEXT,-(R5)    ;PLACE EXTENDED ELAPSED TIME
                                                        ;ARGUMENT ON R5 STACK
000766' 016745  000216                  MOV     TIME,-(R5)      ;PLACE ELAPSED TIME ARGUMENT
                                                        ;ON R5 STACK
000772' 004767  000000G                 JSR     PC,HMS          ;CALL HOURS, MINUTES, SECOND CONVERSION
000776' 012605                          MOV     (SP)+,R5        ;RESTORE R5 STACK POINTER
001000' 012500                          MOV     (R5)+,R0        ;GET OUTPUT ARGUMENT OF HMS
                                                        ;ROUTINE INTO R0
001002' 112067  000155                  MOVB    (R0)+,RUNT1+1   ;LOAD
001006' 112067  000152                  MOVB    (R0)+,RUNT1+2   ;CONVERTED
001012' 112067  000147                  MOVB    (R0)+,RUNT1+3   ;TIME
001016' 112067  000145                  MOVB    (R0)+,RUNT2     ;INTO
001022' 112067  000142                  MOVB    (R0)+,RUNT2+1   ;MESSAGE
001026' 112067  000140                  MOVB    (R0)+,RUNT3
001032' 112067  000135                  MOVB    (R0)+,RUNT3+1
001036' 012605                          MOV     (SP)+,R5        ;RESTORE R5
001040' 012600                          MOV     (SP)+,R0        ;RESTORE R0
001042' 104403  000000' 001112'         MSGN$,BEGIN,EXPRI       ;ASCII MESSAGE CALL WITH COMMON HEADER
001050' 012767  000001  000052  1$:     MOV     #1,RUNING       ;START COUNTING SEC.S FOR PASSES AGAIN
001056' 000167  177354                  JMP     CLKGO
```



```
001062' 012700  001214'         LCLEAR: MOV     #MODTIM,R0      ;GET BEGINING OF TABLE
001066' 012701  001574'                 MOV     #MODEND,R1      ;GET END
001072' 005067  000112                  CLR     TIME            ; ZERO TOTAL RUNTIME
001076' 005067  000110                  CLR     TIMEXT          ; ZERO EXTENDED TIME BITS
001102' 005020                  1$:     CLR     (R0)+           ;CLEAR ENTRY
001104' 020001                          CMP     R0,R1           ;DONE?
001106' 001375                          BNE     1$              ;BR IF NO
001110' 000207                          RTS     PC              ;EXIT

001112' 001142'                 EXPRI:  RUNTI                   ; RUNTIME MESSAGE POINTER
001114' 177777                          177777
001116' 000000                  MSGTIM: OPEN                    ;TELLS WHEN TO TYPE A MSG
001120' 000000                  INTER:  OPEN                    ;HOW OFTEN BETWEEN MSG.S
001122' 000000                  BASE:   OPEN                    ; HOLDS 50. FOR 50HZ, 60. FOR 60HZ
001124' 000000                  CLKCSR: OPEN                    ; HOLDS ADDRESS OF CLOCK'S CSR
001126' 000000                  CLKCTR: OPEN                    ; NUMBER OF INTERUPTS LEFT BETWEEN SECONDS
001130' 000000                  RUNING: OPEN                    ;0 IF BETWEEN PASSES
001132' 000000                  MTIME:  OPEN                    ;COUNTS SECONDS BETWEEN MSG.S
001134' 000000                  TIMCTR: OPEN                    ;COUNTS TICS FOR MSG SECONDS
001136' 000000                  TMPTIM: OPEN                    ; TEMP LOCATION HOLDS TIME
001140' 000000                  HLDTIM: OPEN                    ; TEMP LOCATION HOLDS TIME SINCE LAST EOP

001142' 020045  051040  047125  RUNTI:  .ASCII "%  RUN TIME IS  "
001150' 052040  046511  020105
001156' 051511  020040
001162' 020040  020040      072 RUNT1:  .ASCII "    :"          ; HOURS
001167'     040 035040          RUNT2:  .ASCII "  :"            ; MINUTES
001172' 020040  024040  035110  RUNT3:  .ASCIZ "  (H:M:S)%"     ; SECONDS
001200' 035115  024523  000045

                                ;               DO NOT MODIFY ANY LOCATIONS BELOW THIS POINT.
                                ;               THE MONITOR MUST KNOW EXACTLY WHERE EACH WORD IS.

                                        .EVEN
001206' 000000'                 KW11L:  BEGIN                   ; HOLDS STARTING ADDRESS OF MODULE
001210' 000000                  TIME:   OPEN                    ;ELAPSED TIME IN SECONDS
001212' 000000                  TIMEXT: OPEN                    ; HOLDS EXTENDED BITS FOR THE TIME WORD
001214' 000170                  MODTIM: .BLKW   120.            ;TABLE FOR MONITOR, TWO WORDS FOR EACH MODULE
001574'                         MODEND:


                                ;+
                                ; SET ASIDE R5 STACK STORAGE
                                ;-

001644' 000000                  R5STK:  .WORD   0               ;R5 STACK
        000001                  .END
```

```
ACSR    000102R          180#
ADDR    000006R          146#    221     222
ADDR22= 001000           198#
ASB     000106R          184#
ASTAT   000104R          182#
AWAS    000110R          185#
BASE    001122R          226*    229*    231     234     246     280     350#
BEGIN   000000R          143#    205     236     265     272     276     287     330     371
BIT0  = 000001           198#
BIT1  = 000002           198#
BIT10 = 002000           198#
BIT11 = 004000           198#
BIT12 = 010000           198#
BIT13 = 020000           198#
BIT14 = 040000           198#
BIT15 = 100000           198#
BIT2  = 000004           198#
BIT3  = 000010           198#
BIT4  = 000020           198#
BIT5  = 000040           198#
BIT6  = 000100           198#    235     260     267     286
BIT7  = 000200           198#
BIT8  = 000400           198#
BIT9  = 001000           198#
BREAK$= 104407           198#
BR1     000012R          148#    225
BR2     000013R          149#
BTOD$ = 104421           198#
CDATA$= 104412           198#
CKHUNG= ******  G        200#    270
CLKCSR  001124R          227*    235*    240*    260*    267*    286*    351#
CLKCTR  001126R          230*    238*    352#
CLKGO   000436R          234#    289     332
CLOCK = ******  G        200#    203
CONFIG  000056R          168#
CSRA    000100R          178#
DATCK$= 104411           198#
DATER$= 104404           198#
DVID1   000014R          150#
ENDIT$= 104413           198#    287
ENDS  = 104410           198#    205
ERRTYP  000106R          183#
EXIT$ = 104400           198#    236     272
EXPRI   001112R          330     346#
GETPA$= 104415           198#
GWBUF$= 104414           198#
HLDTIM  001140R          250     253*    285*    357#
HMS   = ******  G        200#    317
HRDCNT  000044R          163#
HRDER$= 104405           198#
HRDPAS  000050R          165#
ICONT   000036R          160#    202*    230*    232
ICOUNT  000040R          161#    282
IDNUM   001122R          190#
IMIT    000030R          157#
INTER   001120R          212#    215*    218*    219     296     349#
```



```
INTR    000120R          189#    232*
KLKINT  000456R          224     240#
KW11L   001206RG         200#    371#
LCLEAR  001062RG         200#    208     336#
MAP22$= 104416           198#
MODEND  001574R          337     375#
MODNAM  000000R          144#
MODSP   000224R          158     196#
MODTIM  001214R          336     374#
MSGHS = 104403           198#    330
MSGS$ = 104402           198#
MSGTIM  001116R          219#    348#
MSG$  = 104401           198#
MTIME   001132R          220*    245*    296     298*    354#
NULL    000000           198#
OPEN  = 000000           145     151     152     153     154     171     172     173     174     175     176     177     178
                         180     182     184     185     187     188     189     198#    348     349     350     351     352
                         353     354     355     356     357     372     373
OTOA$ = 104420           198#
PASCNT  000034R          159#    206     293
PIRQ$ = 000004           198#    265     276
POPSP = 005726           198#
POPSP2= 022626           198#
PRTY  = 000000           142#    198#
PRTY0 = 000000           198#
PRTY1 = 000040           198#
PRTY2 = 000100           198#
PRTY3 = 000140           198#
PRTY4 = 000200           198#
PRTY5 = 000240           198#
PRTY6 = 000300           198#    198#
PRTY7 = 000340           198#
PS    = 177776           198#
PSW   = 177776           198#
PUSH    005746           198#
PUSH2 = 024646           198#
RAND$ = 104417           198#
RANNUM  000054R          167#
REST    000720R          294     296#
RESTRT  000706R          186     290#
RES1    000056R          169#
RES2    000060R          170#
RSTRT   000112R          186#
RUNING  001130R          209*    247     256     284*    331*    353#
RUNTI   001142R          346     359#
RUNT1   001152R          321*    322*    323*    362#
RUNT2   001166R          324*    325*    363#
RUNT3   001172R          326*    327*    364#
R5STK   001644R          269     309     382#
SBADR   000102R          179#
SOFCNT  000042R          162#
SOFER$= 104406           198#
SOFPAS  000046R          164#
SPOINT  000032R          156#
SPSIZ = 000040           191#
SR1     000016R          151#    210     227     299
```

```
SR2     000020R          152#
SR3     000022R          153#
SR4     000024R          154#
START   000224R          157     202#    295
STAT    000026R          156#
SVR0    000062R          171#
SVR1    000064R          172#
SVR2    000066R          173#
SVR3    000070R          174#
SVR4    000072R          175#
SVR5    000074R          176#
SVR6    000076R          177#
SYSCNT  000052R          166#
TIMCTR  001134R          231#    241*    246*    355#
TIME    001210R          243*    249     285     315     338*    372#
TIMEXT  001212R          244*    313     339*    373#
TMPTIM  001136R          249*    250*    251     356#
TRPDFD= 000022           198#
VECTOR  000010R          147#    223
WASADR  000104R          181#
WDFR    000116R          188#
WDTO    000114R          187#
XFLAG   000056R          145#
.       = 001646R        374#

. ABS.  000000      000
        001646      001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 XKWAG0,XKWAG0/SOL/CRF:SYM=DDXCON,XKWAG0
 RUN-TIME:  1 1  .2  SECONDS
 RUN-TIME RATIO: 13/2=4.5
 CORE USED:  7K  (13 PAGES)
```