```
ISIS-II PL/M-86 V2.0 COMPILATION OF MODULE DATE
OBJECT MODULE PLACED IN DATE.OBJ
COMPILER INVOKED BY:  :F0: DATE.PLM XREF OPTIMIZE(3) DEBUG


            $compact
            $title ('DATE: Time and Date utility')
  1         DATE:
            do;

            $include(:f1:copyrt.lit)
        =
        =   /*
        =     Copyright (C) 1983
        =     Digital Research
        =     P.O. Box 579
        =     Pacific Grove, CA 93950
        =   */
        =

            /* Revised:
                23 Jun 82  by Bill Fitler  (CCP/M-86)
            */

            $include(:f1:vaxcmd.lit)
        =
        =   /**** VAX commands for generation - read the name of this program
        =           for PROGNAME below.
        =
        =       $ util := PROGNAME
        =       $ ccpmsetup             ! set up environment
        =       $ assign 'f$directory()' f1:        ! use local dir for temp files
        =       $ plm86 'util'.plm xref 'p1' optimize(3) debug
        =       $ link86 f2:scd.obj, 'util'.obj  to 'util'.lnk
        =       $ loc86 'util'.lnk od(sm(code,dats,data,stack,const)) -
        =            ad(sm(code(0),dats(10000h))) ss(stack(+32)) to 'util'.
        =       $ h86 'util'
        =
        =   ***** Then, on a micro:
        =       A>vax progname.h86 $fans
        =       A>gencmd progname data[b1000]
        =
        =   ***** Notes: Stack is increased for interrupts.  Const(ants) are last
        =           to force hex generation.
        =   ****/

            $include(:f1:vermpm.lit)
        =
        =   /*   This utility requires MP/M or Concurrent function calls */
        =
        =   /****** commented out for CCP/M-86 :
        =   declare Ver$OS literally '11h',
        =       Ver$Needs$OS literally '''Requires MP/M-86''';
        =    ******/
        =
  2   1 =   declare Ver$OS literally '14h',
```

CCP/M-86 2.0 V.4
COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. # CC-104

```
        =       Ver$Needs$OS literally '''Requires Concurrent CP/M-86''';
        =
        =
 3   1  =   declare Ver$Mask literally '0fdh';  /* mask out ls_network bit */
        =
 4   1  =   declare Ver$BDOS literally '30h';   /* minimal BDOS version rqd */
        =

 5   1      declare dcl literally 'declare';
 6   1      dcl lit literally 'literally';
 7   1      dcl forever lit 'while 1';


 8   1        mon1:
                procedure (func,info) external;
 9   2          declare func byte;
10   2          declare info address;
11   2        end mon1;

12   1        mon2:
                procedure (func,info) byte external;
13   2          declare func byte;
14   2          declare info address;
15   2        end mon2;

16   1        mon3:
                procedure (func,info) address external;
17   2          declare func byte;
18   2          declare info address;
19   2        end mon3;

20   1        mon4:
                procedure (func,info) pointer external;
21   2          declare func byte;
22   2          declare info address;
23   2        end mon4;


24   1        declare fcb (1) byte external;
25   1        declare fcb16 (1) byte external;
26   1        declare buff (1) byte external;


27   1        read$console:
                procedure byte;
28   2            return mon2 (1,0);
29   2          end read$console;

30   1        write$console:
                procedure (char);
31   2            declare char byte;
32   2            call mon1 (2,char);
33   2          end write$console;

34   1        print$buffer:
                procedure (buffer$address);
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
35   2            declare buffer$address address;
36   2            call mon1 (9,buffer$address);
37   2          end print$buffer;

38   1        check$console$status:
              procedure byte;
39   2            return mon2 (11,0);
40   2          end check$console$status;

41   1        version:
              procedure address;
42   2            return mon3(12,0);
43   2          end version;

44   1        terminate:
              procedure;
45   2            call mon1 (0,0);
46   2          end terminate;

47   1        get$sysdat:
              procedure pointer;
48   2            return (mon4(154,0));
49   2          end get$sysdat;

50   1        crlf:
              procedure;
51   2            call write$console (0dh);
52   2            call write$console (0ah);
53   2          end crlf;

54   1      error:
            procedure;
55   2          call print$buffer (.(
                'Illegal time/date specification.','$'));
56   2          call terminate;
57   2        end;

          /*******************************************************

                    Time & Date ASCII Conversion Code

           *******************************************************/

58   1      declare tod$adr address;
59   1      declare tod based tod$adr structure (
            opcode byte,
            date address,
            hrs byte,
            min byte,
            sec byte,
            ASCII (21) byte );

60   1      declare string$adr address;
61   1      declare string based string$adr (1) byte;
62   1      declare index byte;
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
 63   1      emitchar: procedure(c);
 64   2          declare c byte;
 65   2          string(index := index + 1) = c;
 66   2          end emitchar;

 67   1      emitn: procedure(a);
 68   2          declare a address;
 69   2          declare c based a byte;
 70   2          do while c <> '$';
 71   3            string(index := index + 1) = c;
 72   3            a = a + 1;
 73   3          end;
 74   2          end emitn;

 75   1      emit$bcd: procedure(b);
 76   2          declare b byte;
 77   2          call emitchar('0'+b);
 78   2          end emit$bcd;

 79   1      emit$bcd$pair: procedure(b);
 80   2          declare b byte;
 81   2          call emit$bcd(shr(b,4));
 82   2          call emit$bcd(b and 0fh);
 83   2          end emit$bcd$pair;

 84   1      emit$colon: procedure(b);
 85   2          declare b byte;
 86   2          call emit$bcd$pair(b);
 87   2          call emitchar(':');
 88   2          end emit$colon;

 89   1      emit$bin$pair: procedure(b);
 90   2          declare b byte;
 91   2          call emit$bcd(b/10);
 92   2          call emit$bcd(b mod 10);
 93   2          end emit$bin$pair;

 94   1      emit$slant: procedure(b);
 95   2          declare b byte;
 96   2          call emit$bin$pair(b);
 97   2          call emitchar('/');
 98   2          end emit$slant;

 99   1      declare chr byte;

100   1      gnc: procedure;
                  /* get next command byte */
101   2          if chr = 0 then return;
103   2          if index = 20 then
104   2          do;
105   3            chr = 0;
106   3            return;
107   3          end;
108   2          chr = string(index := index + 1);
109   2          end gnc;

110   1      deblank: procedure;
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
111   2             do while chr = ' ';
112   3             call gnc;
113   3             end;
114   2         end deblank;

115   1     numeric: procedure byte;
              /* test for numeric */
116   2         return (chr - '0') < 10;
117   2         end numeric;

118   1     scan$numeric: procedure(lb,ub) byte;
119   2         declare (lb,ub) byte;
120   2         declare b byte;
121   2         b = 0;
122   2         call deblank;
123   2         if not numeric then call error;
125   2             do while numeric;
126   3             if (b and 1110$0000b) <> 0 then call error;
128   3             b = shl(b,3) + shl(b,1); /* b = b * 10 */
129   3             if carry then call error;
131   3             b = b + (chr - '0');
132   3             if carry then call error;
134   3             call gnc;
135   3             end;
136   2         if (b < lb) or (b > ub) then call error;
138   2         return b;
139   2         end scan$numeric;

140   1     scan$delimiter: procedure(d,lb,ub) byte;
141   2         declare (d,lb,ub) byte;
142   2         call deblank;
143   2         if chr <> d then call error;
145   2         call gnc;
146   2         return scan$numeric(lb,ub);
147   2         end scan$delimiter;

148   1     declare
              base$year lit '78',   /* base year for computations */
              base$day  lit '0',    /* starting day for base$year 0..6 */
              month$size (*) byte data
              /* jan feb mar apr may jun jul aug sep oct nov dec */
              (   31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31),
              month$days (*) word data
              /* jan feb mar apr may jun jul aug sep oct nov dec */
              (  000,031,059,090,120,151,181,212,243,273,304,334);

149   1     leap$days: procedure(y,m) byte;
150   2         declare (y,m) byte;
              /* compute days accumulated by leap years */
151   2         declare yp byte;
152   2         yp = shr(y,2); /* yp = y/4 */
153   2         if (y and 11b) = 0 and month$days(m) < 59 then
                  /* y not 00, y mod 4 = 0, before march, so not leap yr */
154   2             return yp - 1;
              /* otherwise, yp is the number of accumulated leap days */
155   2         return yp;
156   2         end leap$days;
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
157   1     declare word$value word;

158   1     get$next$digit: procedure byte;
                /* get next lsd from word$value */
159   2         declare lsd byte;
160   2         lsd = word$value mod 10;
161   2         word$value = word$value / 10;
162   2         return lsd;
163   2         end get$next$digit;

164   1     bcd:
            procedure (val) byte;
165   2         declare val byte;
166   2         return shl((val/10),4) + val mod 10;
167   2       end bcd;

168   1     declare (month, day, year, hrs, min, sec) byte;

169   1     set$date$time: procedure;
170   2         declare
                (i, leap$flag) byte; /* temporaries */
171   2         month = scan$numeric(1,12) - 1;
                /* may be feb 29 */
172   2         if (leap$flag := month = 1) then i = 29;
174   2             else i = month$size(month);
175   2         day   = scan$delimiter("/",1,i);
176   2         year  = scan$delimiter("/",base$year,99);
                /* ensure that feb 29 is in a leap year */
177   2         if leap$flag and day = 29 and (year and 11b) <> 0 then
178   2             /* feb 29 of non-leap year */ call error;
                /* compute total days */
179   2          tod.date = month$days(month)
                           + 365 * (year - base$year)
                           + day
                           - leap$days(base$year,0)
                           + leap$days(year,month);

180   2         tod.hrs   = bcd (scan$numeric(0,23));
181   2         tod.min   = bcd (scan$delimiter(":",0,59));
182   2         if tod.opcode = 2 then
                /* date, hours and minutes only */
183   2         do;
184   3           if chr = ":"
                    then i = scan$delimiter (":",0,59);
186   3           tod.sec = 0;
187   3         end;
                /* include seconds */
188   2         else tod.sec   = bcd (scan$delimiter(":",0,59));

189   2         end set$date$time;

190   1     bcd$pair: procedure(a,b) byte;
191   2         declare (a,b) byte;
192   2         return shl(a,4) or b;
193   2         end bcd$pair;
```

```
194  1      compute$year: procedure;
                /* compute year from number of days in word$value */
195  2          declare year$length word;
196  2          year = base$year;
197  2              do forever;
198  3              year$length = 365;
199  3              if (year and 11b) = 0 then /* leap year */
200  3                  year$length = 366;
201  3              if word$value <= year$length then
202  3                  return;
203  3              word$value = word$value - year$length;
204  3              year = year + 1;
205  3              end;
206  2          end compute$year;

207  1      declare
                week$day  byte, /* day of week 0 ... 6 */
                day$list (*) byte data
                ('Sun$Mon$Tue$Wed$Thu$Fri$Sat$'),
                leap$bias byte; /* bias for feb 29 */

208  1      compute$month: procedure;
209  2          month = 12;
210  2              do while month > 0;
211  3              if (month := month - 1) < 2 then /* jan or feb */
212  3                  leapbias = 0;
213  3              if month$days(month) + leap$bias < word$value then return;
215  3              end;
216  2          end compute$month;

217  1      declare
                date$test byte,    /* true if testing date */
                test$value word;   /* sequential date value under test */

218  1      get$date$time: procedure;
                /* get date and time */
219  2          hrs = tod.hrs;
220  2          min = tod.min;
221  2          sec = tod.sec;
222  2          word$value = tod.date;
                /* word$value contains total number of days */
223  2          week$day = (word$value + base$day - 1) mod 7;
224  2          call compute$year;
                /* year has been set, word$value is remainder */
225  2          leap$bias = 0;
226  2          if (year and 11b) = 0 and word$value > 59 then
227  2              /* after feb 29 on leap year */ leap$bias = 1;
228  2          call compute$month;
229  2          day = word$value - (month$days(month) + leap$bias);
230  2          month = month + 1;
231  2          end get$date$time;

232  1      emit$date$time: procedure;
233  2          call emitn(.day$list(shl(week$day,2)));
234  2          call emitchar(' ');
235  2          call emit$slant(month);
236  2          call emit$slant(day);
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
237   2          call emit$bin$pair(year);
238   2          call emitchar(' ');
239   2          call emit$colon(hrs);
240   2          call emit$colon(min);
241   2          call emit$bcd$pair(sec);
242   2          end emit$date$time;

243   1      tod$ASCII:
               procedure (parameter);
244   2          declare parameter address;
245   2          declare ret address;

246   2          ret = 0;
247   2          tod$adr = parameter;
248   2          string$adr = .tod.ASCII;
249   2          if tod.opcode = 0 then
250   2          do;
251   3            call get$date$time;
252   3            index = -1;
253   3            call emit$date$time;
254   3          end;
                 else
255   2          do;
256   3            if (tod.opcode = 1) or
                     (tod.opcode = 2) then
257   3            do;
258   4              chr = string(index:=0);
259   4              call set$date$time;
260   4              ret = .string(index);
261   4            end;
                   else
262   3            do;
263   4              call error;
264   4            end;
265   3          end;
266   2        end tod$ASCII;

             /*************************************************************
              *************************************************************/

267   1        declare tod$pointer pointer;
268   1        declare tod$ptr structure (
                 offset word,
                 segment word) at (@tod$pointer);
269   1        declare extrnl$tod based tod$pointer structure (
                 date address,
                 hrs byte,                /* in system data area */
                 min byte,
                 sec byte );

270   1        declare lcltod structure (        /* local to this program */
                 opcode byte,
                 date address,
                 hrs byte,
                 min byte,
                 sec byte,
                 ASCII (21) byte );
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
271   1       declare i byte;
272   1       declare ret address;

273   1       display$tod:
                procedure;

274   2         lcltod.opcode = 0;        /* read tod */
275   2         call movb (@extrnl$tod.date,@lcltod.date,5);
276   2         call tod$ASCII (.lcltod);
277   2         call write$console (0dh);
278   2         do i = 0 to 20;
279   3           call write$console (lcltod.ASCII(i));
280   3           end;
281   2         end display$tod;


          /*
            Main Program
          */

282   1     declare tod$sd$offset lit '7eh';   /* offset of TOD structure in MP/M-86 */
283   1     declare vers address;
284   1     declare last$dseg$byte byte
            initial (0);

285   1     plmstart:
            procedure public;
286   2         vers = version;
287   2         if low (vers) < Ver$BDOS  or  (high(vers) and Ver$Mask) = 0  then
288   2         do;
289   3            call print$buffer(.(0dh,0ah,Ver$Needs$OS,'$'));
290   3            call mon1(0,0);
291   3         end;

292   2         tod$pointer = get$sysdat;
293   2         tod$ptr.offset = tod$sd$offset;
294   2         if (fcb(1) <> ' ') and (fcb(1) <> 'P') then
295   2         do;
296   3           call move (21,.buff(1),.lcltod.ASCII);
297   3           lcltod.opcode = 1;
298   3           call tod$ASCII (.lcltod);
299   3           call print$buffer (.(
                    'Strike key to set time','$'));
300   3           ret = read$console;
301   3           call movb (@lcltod.date,@extrnl$tod.date,5);  /* use pl/m-86 move */
302   3           call crlf;
303   3         end;
304   2         do while fcb(1) = 'P';
305   3           call display$tod;
306   3           if check$console$status then
307   3           do;
308   4             ret = read$console;
309   4             fcb(1) = 0;
310   4           end;
311   3         end;
312   2         call display$tod;
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
313   2          call terminate;
314   2        end plmstart;

315   1      end DATE;
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

CROSS-REFERENCE LISTING

DEFN	ADDR	SIZE	NAME, ATTRIBUTES, AND REFERENCES									
190	0006H	1	A.	BYTE PARAMETER AUTOMATIC	191	192						
67	0004H	2	A.	WORD PARAMETER AUTOMATIC	68	69	70	71	72			
59	0006H	21	ASCII.	BYTE ARRAY(21) MEMBER(TOD)	248							
270	0006H	21	ASCII.	BYTE ARRAY(21) MEMBER(LCLTOD)	279	296						
75	0004H	1	B.	BYTE PARAMETER AUTOMATIC	76	77						
190	0004H	1	B.	BYTE PARAMETER AUTOMATIC	191	192						
94	0004H	1	B.	BYTE PARAMETER AUTOMATIC	95	96						
120	0016H	1	B.	BYTE 121 126 128 131	136	138						
89	0004H	1	B.	BYTE PARAMETER AUTOMATIC	90	91	92					
84	0004H	1	B.	BYTE PARAMETER AUTOMATIC	85	86						
79	0004H	1	B.	BYTE PARAMETER AUTOMATIC	80	81	82					
148			BASEDAY.	LITERALLY 223								
148			BASEYEAR	LITERALLY 176 179 196								
164	0282H	39	BCD.	PROCEDURE BYTE STACK=0006H	180	181	188					
190	03CAH	17	BCDPAIR.	PROCEDURE BYTE STACK=0006H								
26	0000H	1	BUFF	BYTE ARRAY(1) EXTERNAL(6)	296							
34	0004H	2	BUFFERADDRESS. . .	WORD PARAMETER AUTOMATIC	35	36						
69	0000H	1	C.	BYTE BASED(A) 70 71								
63	0004H	1	C.	BYTE PARAMETER AUTOMATIC	64	65						
			CARRY.	BUILTIN 129 132								
30	0004H	1	CHAR	BYTE PARAMETER AUTOMATIC	31	32						
38	0034H	15	CHECKCONSOLESTATUS	PROCEDURE BYTE STACK=0008H	306							
99	0015H	1	CHR.	BYTE 101 105 108 111	116	131	143	184	258			
208	0410H	59	COMPUTEMONTH . . .	PROCEDURE STACK=0002H	228							
194	03D8H	53	COMPUTEYEAR. . . .	PROCEDURE STACK=0002H	224							
50	006FH	17	CRLF	PROCEDURE STACK=000EH	302							
140	0008H	1	D.	BYTE PARAMETER AUTOMATIC	141	143						
270	0001H	2	DATE	WORD MEMBER(LCLTOD) 275	301							
269	0000H	2	DATE	WORD MEMBER(EXTRNLTOD)	275	301						
1	0002H		DATE	PROCEDURE STACK=0000H								
59	0001H	2	DATE	WORD MEMBER(TOD) 179	222							
217	0023H	1	DATETEST	BYTE								
168	001AH	1	DAY.	BYTE 175 177 179 229	236							
207	0024H	28	DAYLIST.	BYTE ARRAY(28) DATA 233								
5			DCL.	LITERALLY								
110	017CH	17	DEBLANK.	PROCEDURE STACK=0006H	122	142						
273	0567H	74	DISPLAYTOD	PROCEDURE STACK=002EH	305	312						
75	00D3H	16	EMITBCD.	PROCEDURE STACK=000AH	81	82	91	92				
79	00E3H	27	EMITBCDPAIR. . . .	PROCEDURE STACK=0010H	86	241						
89	0111H	39	EMITBINPAIR. . . .	PROCEDURE STACK=0010H	96	237						
63	008FH	28	EMITCHAR	PROCEDURE STACK=0004H	77	87	97	234	238			
84	00FEH	19	EMITCOLON.	PROCEDURE STACK=0016H	239	240						
232	04B5H	77	EMITDATETIME . . .	PROCEDURE STACK=001AH	253							
67	00ABH	40	EMITN.	PROCEDURE STACK=0004H	233							
94	0138H	19	EMITSLANT.	PROCEDURE STACK=0016H	235	236						
54	0080H	15	ERROR.	PROCEDURE STACK=000EH	124	127	130	133	137	144	178	263
269	0000H	5	EXTRNLTOD.	STRUCTURE BASED(TODPOINTER)	275	301						
24	0000H	1	FCB.	BYTE ARRAY(1) EXTERNAL(4)	294	304	309					
25	0000H	1	FCB16.	BYTE ARRAY(1) EXTERNAL(5)								

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
  7                FOREVER. . . . . LITERALLY       197
 20  0000H     1   FUNC . . . . . . BYTE PARAMETER        21
 16  0000H     1   FUNC . . . . . . BYTE PARAMETER        17
 12  0000H     1   FUNC . . . . . . BYTE PARAMETER        13
  8  0000H     1   FUNC . . . . . . BYTE PARAMETER         9
218  0448H   106   GETDATETIME. . . . PROCEDURE STACK=0006H          251
158  0262H    32   GETNEXTDIGIT . . . PROCEDURE BYTE STACK=0002H
 47  0060H    15   GETSYSDAT. . . . . PROCEDURE POINTER STACK=0008H       292
100  014BH    49   GNC. . . . . . . . PROCEDURE STACK=0002H          112  134  145
                   HIGH . . . . . . . BUILTIN       287
270  0003H     1   HRS. . . . . . . . BYTE MEMBER(LCLTOD)
 59  0003H     1   HRS. . . . . . . . BYTE MEMBER(TOD)          180  219
168  001CH     1   HRS. . . . . . . . BYTE       219  239
269  0002H     1   HRS. . . . . . . . BYTE MEMBER(EXTRNLTOD)
271  003FH     1   I. . . . . . . . . BYTE       278  279
170  001FH     1   I. . . . . . . . . BYTE       173  174  175  185
 62  0014H     1   INDEX. . . . . . . BYTE        65   71  103  108  252  258  260
 20  0000H     2   INFO . . . . . . . WORD PARAMETER       22
 16  0000H     2   INFO . . . . . . . WORD PARAMETER       18
 12  0000H     2   INFO . . . . . . . WORD PARAMETER       14
  8  0000H     2   INFO . . . . . . . WORD PARAMETER       10
284  0040H     1   LASTDSEGBYTE . . . BYTE INITIAL
118  0006H     1   LB . . . . . . . . BYTE PARAMETER AUTOMATIC       119  136
140  0006H     1   LB . . . . . . . . BYTE PARAMETER AUTOMATIC       141  146
270  0024H    27   LCLTOD . . . . . . STRUCTURE       274  275  276  279  296  297  298  301
207  0022H     1   LEAPBIAS . . . . . BYTE       212  213  225  227  229
149  0236H    44   LEAPDAYS . . . . . PROCEDURE BYTE STACK=0006H       179
170  0020H     1   LEAPFLAG . . . . . BYTE       172  177
  6                LIT. . . . . . . . LITERALLY       7  148  282
                   LOW. . . . . . . . BUILTIN       287
159  0018H     1   LSD. . . . . . . . BYTE       160  162
149  0004H     1   M. . . . . . . . . BYTE PARAMETER AUTOMATIC       150  153
269  0003H     1   MIN. . . . . . . . BYTE MEMBER(EXTRNLTOD)
 59  0004H     1   MIN. . . . . . . . BYTE MEMBER(TOD)          181  220
270  0004H     1   MIN. . . . . . . . BYTE MEMBER(LCLTOD)
168  001DH     1   MIN. . . . . . . . BYTE       220  240
  8  0000H         MON1 . . . . . . . PROCEDURE EXTERNAL(0) STACK=0000H        32   36   45  290
 12  0000H         MON2 . . . . . . . PROCEDURE BYTE EXTERNAL(1) STACK=0000H        28   39
 16  0000H         MON3 . . . . . . . PROCEDURE WORD EXTERNAL(2) STACK=0000H        42
 20  0000H         MON4 . . . . . . . PROCEDURE POINTER EXTERNAL(3) STACK=0000H        48
168  0019H     1   MONTH. . . . . . . BYTE       171  172  174  179  209  210  211  213  229  230  235
148  0000H    24   MONTHDAYS. . . . . WORD ARRAY(12) DATA       153  179  213  229
148  0018H    12   MONTHSIZE. . . . . BYTE ARRAY(12) DATA       174
                   MOVB . . . . . . . BUILTIN       275  301
                   MOVE . . . . . . . BUILTIN       296
115  018DH    17   NUMERIC. . . . . . PROCEDURE BYTE STACK=0002H       123  125
268  0000H     2   OFFSET . . . . . . WORD MEMBER(TODPTR)      293
 59  0000H     1   OPCODE . . . . . . BYTE MEMBER(TOD)          182  249  256
270  0000H     1   OPCODE . . . . . . BYTE MEMBER(LCLTOD)       274  297
243  0004H     2   PARAMETER. . . . . WORD PARAMETER AUTOMATIC       244  247
285  05B1H   170   PLMSTART . . . . . PROCEDURE PUBLIC STACK=0032H
 34  0024H    16   PRINTBUFFER. . . . PROCEDURE STACK=000AH        55  289  299
 27  0002H    15   READCONSOLE. . . . PROCEDURE BYTE STACK=0008H       300  308
272  0010H     2   RET. . . . . . . . WORD       300  308
245  000AH     2   RET. . . . . . . . WORD       246  260
140  0215H    33   SCANDELIMITER. . . PROCEDURE BYTE STACK=0020H       175  176  181  185  188
118  019EH   119   SCANNUMERIC. . . . PROCEDURE BYTE STACK=0016H       146  171  180
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
 59  0005H     1  SEC. . . . . .   BYTE MEMBER(TOD)          136  188  221
270  0005H     1  SEC. . . . . .   BYTE MEMBER(LOCLTOD)
269  0004H     1  SEC. . . . . .   BYTE MEMBER(EXTRNLTOD)
168  001EH     1  SEC. . . . . .   BYTE       221  241
268  0002H     2  SEGMENT. . . . .  WORD MEMBER(TODPTR)
169  02A9H   289  SETDATETIME. . .  PROCEDURE STACK=0024H          259
                 SHL. . . . . .   BUILTIN          128  166  192  233
                 SHR. . . . . .   BUILTIN           81  152
 61  0000H     1  STRING . . . . .  BYTE BASED(STRINGADR) ARRAY(1)      65   71  108  258  260
 60  0002H     2  STRINGADR. . . .  WORD        61   65   71  108  248  258  260
 44  0052H    14  TERMINATE. . . .  PROCEDURE STACK=0008H           56  313
217  0008H     2  TESTVALUE. . . .  WORD
 59  0000H    27  TOD. . . . . .   STRUCTURE BASED(TODADR)       179  180  181  182  185  188  219  220
                  221  222  248  249  256
 58  0000H     2  TODADR . . . . .  WORD         59  179  180  181  182  186  188  219  220  221  222  247
                  248  249  256
243  0502H   101  TODASCII . . . .  PROCEDURE STACK=002AH          276  298
267  000CH     4  TODPOINTER . . .  POINTER         268  269  275  292  301
268  000CH     4  TODPTR . . . . .  STRUCTURE AT          293
282              TODSOOFFSET. . .  LITERALLY        293
118  0004H     1  UB . . . . . .   BYTE PARAMETER AUTOMATIC      119  136
140  0004H     1  UB . . . . . .   BYTE PARAMETER AUTOMATIC      141  146
164  0004H     1  VAL. . . . . .   BYTE PARAMETER AUTOMATIC      165  166
  4              VERBDOS. . . . .  LITERALLY        287
  3              VERMASK. . . . .  LITERALLY        287
  2              VERNEEDSOS . . .  LITERALLY        289
  2              VEROS. . . . . .  LITERALLY
283  0012H     2  VERS . . . . . .  WORD        286  287
 41  0043H    15  VERSION. . . . .  PROCEDURE WORD STACK=0008H          286
207  0021H     1  WEEKDAY. . . . .  BYTE        223  233
157  0004H     2  WORDVALUE. . . .  WORD        160  161  201  203  213  222  223  226  229
 30  0011H    19  WRITECONSOLE . .  PROCEDURE STACK=000AH           51   52  277  279
149  0006H     1  Y. . . . . .     BYTE PARAMETER AUTOMATIC      150  152  153
168  0018H     1  YEAR . . . . . .  BYTE        176  177  179  196  199  204  226  237
195  0006H     2  YEARLENGTH . . .  WORD        198  200  201  203
151  0017H     1  YP . . . . . .   BYTE        152  154  155
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

MODULE INFORMATION:

```
  CODE AREA SIZE     = 065BH   1627D
  CONSTANT AREA SIZE = 0096H    150D
  VARIABLE AREA SIZE = 0041H     65D
  MAXIMUM STACK SIZE = 0032H     50D
  511 LINES READ
  0 PROGRAM ERROR(S)
```

END OF PL/M-86 COMPILATION

```
ISIS-II MCS-86 LOCATER, V1.2 INVOKED BY:
:F0: DATE.LNK OD(SM(CODE,DATS,DATA,STACK,CONST)) AD(SM(CODE(0),DATS(10000H))) SS(STACK(+32)) TO DATE.

SYMBOL TABLE OF MODULE SCD
READ FROM FILE DATE.LNK
WRITTEN TO FILE DATE.

BASE   OFFSET TYPE SYMBOL                BASE   OFFSET TYPE SYMBOL

0000H  06B1H  PUB  PLMSTART              0000H  005DH  PUB  XDOS
0000H  005DH  PUB  MON1                  0000H  005DH  PUB  MON2
0000H  005DH  PUB  MON3                  0000H  005DH  PUB  MON4
1000H  0004H  PUB  BDISK                 1000H  0006H  PUB  MAXB
1000H  0050H  PUB  CMDRV                 1000H  0051H  PUB  PASS0
1000H  0053H  PUB  LEN0                  1000H  0054H  PUB  PASS1
1000H  0056H  PUB  LEN1                  1000H  005CH  PUB  FCB
1000H  006CH  PUB  FCB16                 1000H  007CH  PUB  CR
1000H  007DH  PUB  RR                    1000H  007FH  PUB  RO
1000H  0080H  PUB  BUFF                  1000H  0080H  PUB  TBUFF
1000H  0080H  PUB  BUFFA                 1000H  005CH  PUB  FCBA
1000H  0100H  PUB  SAVEAX                1000H  0102H  PUB  SAVEBX
1000H  0104H  PUB  SAVECX                1000H  0106H  PUB  SAVEDX

DATE:  SYMBOLS AND LINES
1000H  0240H  SYM  MEMORY                0000H  0102H  SYM  READCONSOLE
0000H  0111H  SYM  WRITECONSOLE          STACK  0004H  SYM  CHAR
0000H  0124H  SYM  PRINTBUFFER           STACK  0004H  SYM  BUFFERADDRESS
0000H  0134H  SYM  CHECKCONSOLESTAT      0000H  0143H  SYM  VERSION
                   -US
0000H  0152H  SYM  TERMINATE             0000H  0160H  SYM  GETSYSDAT
0000H  016FH  SYM  CRLF                  0000H  0180H  SYM  ERROR
1000H  0108H  SYM  TODADR                1000H  0108H  BAS  TOD
1000H  010AH  SYM  STRINGADR             1000H  010AH  BAS  STRING
1000H  011CH  SYM  INDEX                 0000H  018FH  SYM  EMITCHAR
STACK  0004H  SYM  C                     0000H  01ABH  SYM  EMITN
STACK  0004H  SYM  A                     STACK  0004H  BAS  C
0000H  01D3H  SYM  EMITBCD               STACK  0004H  SYM  B
0000H  01E3H  SYM  EMITBCDPAIR           STACK  0004H  SYM  B
0000H  01FEH  SYM  EMITCOLON             STACK  0004H  SYM  B
0000H  0211H  SYM  EMITBINPAIR           STACK  0004H  SYM  B
0000H  0238H  SYM  EMITSLANT             STACK  0004H  SYM  B
1000H  011DH  SYM  CHR                   0000H  024BH  SYM  GNC
0000H  027CH  SYM  DEBLANK               0000H  028DH  SYM  NUMERIC
0000H  029EH  SYM  SCANNUMERIC           STACK  0006H  SYM  LB
STACK  0004H  SYM  UB                    1000H  011EH  SYM  B
0000H  0315H  SYM  SCANDELIMITER         STACK  0008H  SYM  D
STACK  0006H  SYM  LB                    STACK  0004H  SYM  UB
1000H  01B4H  SYM  MONTHSIZE             1000H  019CH  SYM  MONTHDAYS
0000H  0336H  SYM  LEAPDAYS             STACK  0006H  SYM  Y
STACK  0004H  SYM  M                     1000H  011FH  SYM  YP
1000H  010CH  SYM  WORDVALUE             0000H  0362H  SYM  GETNEXTDIGIT
1000H  0120H  SYM  LSD                   0000H  0382H  SYM  BCD
STACK  0004H  SYM  VAL                   1000H  0121H  SYM  MONTH
1000H  0122H  SYM  DAY                   1000H  0123H  SYM  YEAR
1000H  0124H  SYM  HRS                   1000H  0125H  SYM  MIN
1000H  0126H  SYM  SEC                   0000H  03A9H  SYM  SETDATETIME
1000H  0127H  SYM  I                     1000H  0128H  SYM  LEAPFLAG
0000H  04CAH  SYM  BCDPAIR               STACK  0006H  SYM  A
STACK  0004H  SYM  B                     0000H  04DBH  SYM  COMPUTEYEAR
1000H  010EH  SYM  YEARLENGTH            1000H  0129H  SYM  WEEKDAY
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
1000H  01C0H  SYM  DAYLIST
0000H  0510H  SYM  COMPUTEMONTH
1000H  0110H  SYM  TESTVALUE
0000H  0585H  SYM  EMITDATETIME
STACK  0004H  SYM  PARAMETER
1000H  0114H  SYM  TODPOINTER
1000H  0114H  BAS  EXTRNLTOD
1000H  0147H  SYM  I
0000H  0667H  SYM  DISPLAYTOD
1000H  0148H  SYM  LASTDSEGBYTE
0000H  0102H  LIN     27
0000H  0111H  LIN     29
0000H  0114H  LIN     32
0000H  0124H  LIN     34
0000H  0130H  LIN     37
0000H  0137H  LIN     39
0000H  0143H  LIN     41
0000H  0152H  LIN     43
0000H  0155H  LIN     45
0000H  0160H  LIN     47
0000H  016FH  LIN     49
0000H  0172H  LIN     51
0000H  017EH  LIN     53
0000H  0183H  LIN     55
0000H  018DH  LIN     57
0000H  0192H  LIN     65
0000H  01ABH  LIN     67
0000H  01B6H  LIN     71
0000H  01CDH  LIN     73
0000H  01D3H  LIN     75
0000H  01DFH  LIN     78
0000H  01E6H  LIN     81
0000H  01FAH  LIN     83
0000H  0201H  LIN     86
0000H  020DH  LIN     88
0000H  0214H  LIN     91
0000H  0234H  LIN     93
0000H  023BH  LIN     96
0000H  0247H  LIN     98
0000H  024EH  LIN    101
0000H  0257H  LIN    103
0000H  0263H  LIN    106
0000H  027AH  LIN    109
0000H  027FH  LIN    111
0000H  0289H  LIN    113
0000H  028DH  LIN    115
0000H  029EH  LIN    117
0000H  02A1H  LIN    121
0000H  02A9H  LIN    123
0000H  02B5H  LIN    125
0000H  02C3H  LIN    127
0000H  02D8H  LIN    129
0000H  02E4H  LIN    131
0000H  02F6H  LIN    133
0000H  02FCH  LIN    135
0000H  030BH  LIN    137
0000H  0315H  LIN    139
0000H  0318H  LIN    142
0000H  0323H  LIN    144
0000H  0329H  LIN    146

1000H  012AH  SYM  LEAPBITS
1000H  012BH  SYM  DATETEST
0000H  0546H  SYM  GETDATETIME
0000H  0602H  SYM  TODASCII
1000H  0112H  SYM  RET
1000H  0114H  SYM  TODPTR
1000H  012CH  SYM  LCLTOD
1000H  0118H  SYM  RLT
1000H  011AH  SYM  VERS
0000H  06B1H  SYM  PLMSTART
0000H  0105H  LIN     28
0000H  0111H  LIN     30
0000H  0120H  LIN     33
0000H  0127H  LIN     36
0000H  0134H  LIN     38
0000H  0143H  LIN     40
0000H  0146H  LIN     42
0000H  0152H  LIN     44
0000H  015EH  LIN     46
0000H  0163H  LIN     48
0000H  016FH  LIN     50
0000H  0178H  LIN     52
0000H  0180H  LIN     54
0000H  018AH  LIN     56
0000H  018FH  LIN     63
0000H  01A7H  LIN     66
0000H  01AEH  LIN     70
0000H  01CAH  LIN     72
0000H  01CFH  LIN     74
0000H  01D6H  LIN     77
0000H  01E3H  LIN     79
0000H  01F1H  LIN     82
0000H  01FEH  LIN     84
0000H  0207H  LIN     87
0000H  0211H  LIN     89
0000H  0224H  LIN     92
0000H  0238H  LIN     94
0000H  0241H  LIN     97
0000H  024BH  LIN    100
0000H  0255H  LIN    102
0000H  025EH  LIN    105
0000H  0265H  LIN    108
0000H  027CH  LIN    110
0000H  0286H  LIN    112
0000H  028BH  LIN    114
0000H  0290H  LIN    116
0000H  029EH  LIN    118
0000H  02A6H  LIN    122
0000H  02B2H  LIN    124
0000H  02BCH  LIN    126
0000H  02C6H  LIN    128
0000H  02E1H  LIN    130
0000H  02EDH  LIN    132
0000H  02F9H  LIN    134
0000H  02FEH  LIN    136
0000H  030EH  LIN    138
0000H  0315H  LIN    140
0000H  031BH  LIN    143
0000H  0326H  LIN    145
0000H  0336H  LIN    147
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
0000H  0336H  LIN   149
0000H  0343H  LIN   153
0000H  0358H  LIN   155
0000H  0362H  LIN   158
0000H  0373H  LIN   161
0000H  0382H  LIN   163
0000H  0385H  LIN   166
0000H  03A9H  LIN   169
0000H  03BAH  LIN   172
0000H  03D2H  LIN   174
0000H  03EFH  LIN   176
0000H  0413H  LIN   178
0000H  045DH  LIN   180
0000H  0488H  LIN   182
0000H  0498H  LIN   185
0000H  04AFH  LIN   187
0000H  04C8H  LIN   189
0000H  04CDH  LIN   192
0000H  04D8H  LIN   194
0000H  04E3H  LIN   197
0000H  04E9H  LIN   199
0000H  04F6H  LIN   201
0000H  0501H  LIN   203
0000H  050CH  LIN   205
0000H  0510H  LIN   208
0000H  0518H  LIN   210
0000H  052BH  LIN   212
0000H  0547H  LIN   214
0000H  0549H  LIN   216
0000H  054EH  LIN   219
0000H  055EH  LIN   221
0000H  056AH  LIN   223
0000H  0579H  LIN   225
0000H  058CH  LIN   227
0000H  0594H  LIN   229
0000H  05B3H  LIN   231
0000H  05B8H  LIN   233
0000H  05D0H  LIN   235
0000H  05DEH  LIN   237
0000H  05EBH  LIN   239
0000H  05F9H  LIN   241
0000H  0602H  LIN   243
0000H  060BH  LIN   247
0000H  061AH  LIN   249
0000H  0622H  LIN   252
0000H  062AH  LIN   254
0000H  063AH  LIN   258
0000H  064EH  LIN   260
0000H  0660H  LIN   263
0000H  0667H  LIN   273
0000H  066FH  LIN   275
0000H  068AH  LIN   277
0000H  069CH  LIN   279
0000H  06AFH  LIN   281
0000H  06B4H  LIN   286
0000H  06CAH  LIN   289
0000H  06DAH  LIN   292
0000H  06EBH  LIN   294
0000H  0707H  LIN   297
0000H  0713H  LIN   299
0000H  0339H  LIN   152
0000H  0357H  LIN   154
0000H  0362H  LIN   156
0000H  0365H  LIN   160
0000H  037DH  LIN   162
0000H  0382H  LIN   164
0000H  03A9H  LIN   167
0000H  03ACH  LIN   171
0000H  03CBH  LIN   173
0000H  03DFH  LIN   175
0000H  03FEH  LIN   177
0000H  0416H  LIN   179
0000H  0471H  LIN   181
0000H  0491H  LIN   184
0000H  04A7H  LIN   186
0000H  04B1H  LIN   188
0000H  04CAH  LIN   190
0000H  04D8H  LIN   193
0000H  04DEH  LIN   196
0000H  04E3H  LIN   198
0000H  04F0H  LIN   200
0000H  04FFH  LIN   202
0000H  0508H  LIN   204
0000H  050EH  LIN   206
0000H  0513H  LIN   209
0000H  051FH  LIN   211
0000H  0530H  LIN   213
0000H  0549H  LIN   215
0000H  054BH  LIN   218
0000H  0558H  LIN   220
0000H  0564H  LIN   222
0000H  0576H  LIN   224
0000H  057EH  LIN   226
0000H  0591H  LIN   228
0000H  05AFH  LIN   230
0000H  05B5H  LIN   232
0000H  05CAH  LIN   234
0000H  05D7H  LIN   236
0000H  05E5H  LIN   238
0000H  05F2H  LIN   240
0000H  0600H  LIN   242
0000H  0605H  LIN   246
0000H  0611H  LIN   248
0000H  061FH  LIN   251
0000H  0627H  LIN   253
0000H  062CH  LIN   256
0000H  064BH  LIN   259
0000H  065EH  LIN   261
0000H  0663H  LIN   266
0000H  066AH  LIN   274
0000H  0683H  LIN   276
0000H  0690H  LIN   278
0000H  06A9H  LIN   280
0000H  0681H  LIN   285
0000H  06BAH  LIN   287
0000H  06D1H  LIN   290
0000H  06E5H  LIN   293
0000H  06F9H  LIN   296
0000H  070CH  LIN   298
0000H  071AH  LIN   300
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
0000H  0722H  LIN     301          0000H  0730H  LIN     302
0000H  0733H  LIN     304          0000H  073AH  LIN     305
0000H  073DH  LIN     306          0000H  0744H  LIN     308
0000H  074CH  LIN     309          0000H  0751H  LIN     311
0000H  0753H  LIN     312          0000H  0756H  LIN     313
0000H  0759H  LIN     314          0000H  0102H  LIN     315

MEMORY MAP OF MODULE SCD
READ FROM FILE DATE.LNK
WRITTEN TO FILE DATE.

SEGMENT MAP

START     STOP      LENGTH ALIGN NAME            CLASS

00000H    0075AH    075BH   G    CODE            CODE
10000H    10107H    0108H   G    DATS            DATA
10108H    10148H    0041H   W    DATA            DATA
1014AH    10198H    0052H   W    STACK           STACK
1019CH    10231H    0096H   W    CONST           CONST
10240H    10240H    0000H   G    ??SEG
10240H    10240H    0000H   W    MEMORY          MEMORY


GROUP MAP

ADDRESS   GROUP OR SEGMENT NAME
00000H    CGROUP
          CODE
10000H    DGROUP
          DATS
          STACK
          CONST
          DATA
```

COPYRIGHT © 1981, 1982, 1983
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____