```
;Configuration program for RTC os S-100 PIC/RTC Board
;This program initilizes allows the configuration of the MM58167 Clock chip.
;
;     John Monahan      S100Computers.com 4/23/2010
;
;Note there are two ways of utilizing the clock chip:-
;     1. As a simple clock utilizing all registers and chip RAM for time and date
display.
;     2. Utilizing the chip registers/RAM for time, but allowing CPM3 to utilize the
chip
;        date registers/RAM for date calculation and storage.
;
;
;     PORT ASSIGNMENTS
;
RTCSEL            EQU   0A0H  ; RTC Reg Select of MM58167 chip (Write only)
RTCDATA           EQU   0A1H  ; RTC Data

;RTCSEL           EQU   15H   ;SD Systems SIO8 board, RTC Reg Select of MM58167 chip
;RTCDATA          EQU   16H   ;SD Systems SIO8 board, RTC Data
;
; BDOS EQUATES (VERSION 2)
;
RDCON       EQU   1     ;CP/M Read character
WRCON       EQU   2     ;CP/M Write character
CONSTAT           EQU   11    ;CP/M get consol status
PRINT       EQU   9     ;CP/M Print string
BDOS        EQU   5
ESC         EQU   1BH
CR          EQU   0DH
LF          EQU   0AH

      ORG   100H
START:
      LD    SP,STACK
;
      LD    DE,SIGNON   ;Signon/main menu,RTC Register Select Port
      LD    C,PRINT
      CALL  BDOS
      LD    A,RTCSEL
      CALL  LBYTE

      LD    DE,SIGNON2  ;RTC Data Port =
      LD    C,PRINT
      CALL  BDOS
      LD    A,RTCDATA
      CALL  LBYTE

      LD    DE,SIGNON3  ;CR/LF
      LD    C,PRINT
      CALL  BDOS

      LD    A,0FFH
      LD    (CPM3_FLAG),A     ;start with CPM3 mode ON (i.e. date registers/RAM
protected).

;
;---------------- Configure the MM58167 Clock chip -------------------------
;
BEGINTIME:
      LD    DE,TIMEMENU ;Time menu
      LD    C,PRINT
      CALL  BDOS
;
      LD    A,(CPM3_FLAG)
      OR    A,A         ;0 if currently OFF
```

```
      JP    Z,MENU1
      LD    DE,TIMEMENU2      ;Turn on CPM3 flag
      JP    MENU2
MENU1:      LD    DE,TIMEMENU1      ;Turn on CPM3 flag
MENU2:      LD    C,PRINT
      CALL  BDOS

      LD    C,RDCON           ;Get a char
      CALL  BDOS
      PUSH  AF
      LD    DE,CRLFMSG  ;Finish Line
      LD    C,PRINT
      CALL  BDOS
      POP   AF
;
      CP    ESC
      JP    Z,DONE
      CP    30H         ;Show Time & Date
      JP    Z,SHOW_TIME
      CP    31H         ;Get Time information
      JP    Z,GTIME
      CP    32H
      JP    Z,STIME           ;Set Time
      CP    33H
      JP    Z,ZEROTIME  ;Zero Seconds
      CP    34H
      JP    Z,DUMP_REGISTERS ;Dump (raw) core registers
      CP    35H
      JP    Z,INT_LOOP
      CP    36H
      JP    Z,TOGGLE_CPM3
;
AGAIN:      LD    DE,ABORTMSG
      LD    C,PRINT
      CALL  BDOS
      JP    BEGINTIME
;
DONE:
      JP    0H
;
TOGGLE_CPM3:                  ;Toggle CPM3 flag on/off
      LD    A,(CPM3_FLAG)
      OR    A,A
      JP    NZ,TOGGLE1
      LD    A,0FFH
      LD    (CPM3_FLAG),A
      LD    DE,CPM3_OFF_MSG
      LD    C,PRINT
      CALL  BDOS
      JP    BEGINTIME
TOGGLE1:
      XOR   A,A
      LD    (CPM3_FLAG),A
      LD    DE,CPM3_ON_MSG
      LD    C,PRINT
      CALL  BDOS
      JP    BEGINTIME

;
;<<<<<<<<<<<<<<<<< GET THE TIME STRING
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;
SHOW_TIME:
      LD    DE,TIME_MSG
      LD    C,PRINT
      CALL  BDOS
```

```
      LD    A,04H
      OUT   (RTCSEL),A  ;Point to hours
      IN    A,(RTCDATA)
      CALL  PRINTREG    ;Display value in decimal

      LD    E,':'
      LD    C,WRCON           ;Write to consol
      CALL  BDOS

      LD    A,03H
      OUT   (RTCSEL),A  ;Point to minutes
      IN    A,(RTCDATA)
      CALL  PRINTREG    ;Display value in decimal

      LD    E,':'
      LD    C,WRCON           ;Write to consol
      CALL  BDOS

      LD    A,02H
      OUT   (RTCSEL),A  ;Point to seconds
      IN    A,(RTCDATA)
      CALL  PRINTREG    ;Display value in decimal

      LD    DE,DATE_MSG ;Date=
      LD    C,PRINT
      CALL  BDOS

      LD    A,(CPM3_FLAG)
      OR    A,A
      JP    NZ,STIME1   ;If CPM3 flag skip direct register information

      LD    A,07H
      OUT   (RTCSEL),A  ;Point to Months
      IN    A,(RTCDATA)
      CALL  PRINTREG    ;Display value in decimal

      LD    E,'/'
      LD    C,WRCON           ;Write to consol
      CALL  BDOS

      LD    A,06H
      OUT   (RTCSEL),A  ;Point to DOM
      IN    A,(RTCDATA)
      CALL  PRINTREG    ;Display value in decimal

      LD    DE,YEAR_MSG ;"/20"
      LD    C,PRINT
      CALL  BDOS

      LD    A,09H
      OUT   (RTCSEL),A  ;1/100 & 1/10 sec, Point to "Year Store"
      IN    A,(RTCDATA)
      CALL  PRINTREG    ;Display value in decimal
      JP    GTIME8            ;All done
;
STIME1:     LD    A,0CH       ;If CPM3 then calculate date information from CPM store

      OUT   (RTCSEL),A  ;CPM3 RAM store. This is where the CPM3 clock
      IN    A,(RTCDATA) ;stores the 16 bit days count (HIGH Byte)
      LD    H,A

      LD    A,0BH
      OUT   (RTCSEL),A  ;CPM3 RAM store. (LOW Byte of 16 bit days count)
      IN    A,(RTCDATA)
      LD    L,A
```

```
      CALL  C3DMY       ;Convert a CPM3 day count in HL to D=day E=month HL=year

      PUSH  HL
      PUSH  DE

      LD    D,0
      CALL  PRINT_DEC   ;DE -> Ascii decimal string

      LD    E,'/'
      LD    C,WRCON           ;Write to consol
      CALL  BDOS

      POP   DE
      LD    E,D
      LD    D,0
      CALL  PRINT_DEC   ;DE -> Ascii decimal string

      LD    E,'/'
      LD    C,WRCON           ;Write to consol
      CALL  BDOS

      POP   DE
      CALL  PRINT_DEC   ;DE -> Ascii decimal string
      JP    GTIME8
;
;
;<<<<<<<<<<<<<<<< GET THE TIME & DAT INFORMATION
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;
GTIME:      LD    A,(CPM3_FLAG)           ;Get Time
      OR    A,A
      JP    Z,GTIME2
      LD    DE,DATA_MSG3
      LD    C,PRINT
      CALL  BDOS
      JP    GTIME3
GTIME2:     LD    DE,DATA_MSG
      LD    C,PRINT
      CALL  BDOS

GTIME3:     LD    A,02H
      OUT   (RTCSEL),A        ;Point to seconds
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,PORT_SECS_MSG
      LD    C,PRINT
      CALL  BDOS

      LD    A,03H
      OUT   (RTCSEL),A        ;Point to minutes
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,PORT_MINS_MSG
      LD    C,PRINT
      CALL  BDOS

      LD    A,04H
      OUT   (RTCSEL),A        ;Point to hours
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,PORT_HRS_MSG
      LD    C,PRINT
      CALL  BDOS

      LD    A,05H
```

```
      OUT   (RTCSEL),A        ;Point to DOW
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,PORT_DOW_MSG
      LD    C,PRINT
      CALL  BDOS

      LD    A,06H
      OUT   (RTCSEL),A        ;Point to DOM
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,PORT_DOM_MSG         ;Note CPM3 the clock device driver will set this
to 1. Chip will update extra days
      LD    C,PRINT                 ;if CPM3 is not running. Rebooting CPM resets it
to 1 and updates days count below
      CALL  BDOS

      LD    A,07H
      OUT   (RTCSEL),A        ;Point to Months
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,PORT_MONTH_MSG ;Note CPM3the clock device driver will set this to 1.
Chip will update extra months
      LD    C,PRINT                 ;if CPM3 is not running. Rebooting CPM resets it
to 1 and updates days count below
      CALL  BDOS

      LD    A,09H
      OUT   (RTCSEL),A        ;1/100 & 1/10 sec, Point to "Year Store" (Not used by
CPM3 driver, see below)
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,RAM_YEAR_STORE_MSG
      LD    C,PRINT
      CALL  BDOS

      LD    A,(CPM3_FLAG)
      OR    A,A
      JP    Z,GTIME8          ;If no CPM3 flag skip next the rest of chip RAM
locations

      LD    A,0BH
      OUT   (RTCSEL),A        ;Point to minutes in RAM (Actully CPM days count LOW
byte)
      IN    A,(RTCDATA)
      CALL  LBYTE             ;Display value in HEX
      LD    DE,RAM_MINS_MSG3
      LD    C,PRINT
      CALL  BDOS

      LD    A,0CH
      OUT   (RTCSEL),A        ;Point to Hours in RAM (Actully CPM days count HIGH
byte)
      IN    A,(RTCDATA)
      CALL  LBYTE             ;Display value in HEX
      LD    DE,RAM_HRS_MSG3
      LD    C,PRINT
      CALL  BDOS

      LD    A,0DH
      OUT   (RTCSEL),A        ;Point to DOW in RAM
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,RAM_DOW_MSG
      LD    C,PRINT
      CALL  BDOS
```

```
      LD    A,0EH
      OUT   (RTCSEL),A        ;Point to DOM in RAM (CPM days since last reboot)
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,RAM_DOM_MSG3
      LD    C,PRINT
      CALL  BDOS

      LD    A,0EH
      OUT   (RTCSEL),A        ;Point to Month in RAM  (CPM months since last reboot)
      IN    A,(RTCDATA)
      CALL  PRINTREG          ;Display value in decimal
      LD    DE,RAM_MONTH_MSG3
      LD    C,PRINT
      CALL  BDOS
;
      LD    A,0CH
      OUT   (RTCSEL),A        ;Go back to Hours store again. This is where the CPM3
clock
      IN    A,(RTCDATA)       ;stores the 16 bit days count (HIGH Byte)
      LD    H,A
      PUSH  HL                ;store for now
      CALL  LBYTE

      LD    A,0BH
      OUT   (RTCSEL),A        ;Point to Minute store for (LOW Byte of 16 bit days
count)
      IN    A,(RTCDATA)
      POP   HL
      LD    L,A
      PUSH  HL                ;Store for now, Will use again shortly
      CALL  LBYTE

      LD    DE,CPM_DAYS_MSG         ;"CPM Store of days since 1/1/1978"
      LD    C,PRINT
      CALL  BDOS

      LD    DE,CPM_CALCULATED_DATE_MSG    ;"Calculated CPM date ("
      LD    C,PRINT
      CALL  BDOS

      POP   HL                ;<--- HL contains the 16 bit CPM days count
      PUSH  HL
      PUSH  HL
      LD    A,H
      CALL  LBYTE             ;Show (CPM total days again in hex)
      POP   HL
      LD    A,L
      CALL  LBYTE

      LD    DE,BRACKET_MSG          ;") = "
      LD    C,PRINT
      CALL  BDOS

      POP   HL                ;Now convert to real decimal values

      CALL  C3DMY             ;Convert a CPM3 day count in HL to D=day E=month HL=year

      PUSH  HL
      PUSH  DE

      LD    D,0
      CALL  PRINT_DEC         ;DE -> Ascii decimal string

      LD    E,'/'
```

```
      LD    C,WRCON                 ;Write to consol
      CALL  BDOS

      POP   DE
      LD    E,D
      LD    D,0
      CALL  PRINT_DEC         ;DE -> Ascii decimal string

      LD    E,'/'
      LD    C,WRCON                 ;Write to consol
      CALL  BDOS

      POP   DE
      CALL  PRINT_DEC         ;DE -> Ascii decimal string

GTIME8:     LD    DE,CRLFMSG
      LD    C,PRINT
      CALL  BDOS
      JP    BEGINTIME         ;We are done
;
;
;   <<<<<<<<<<<< SET THE TIME >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;
;   NOTE: Will NOT change DATE registers/RAM if CPM3 flag is set because the CPM clock
device driver
;   stores the 16 bit Days count in the chips RAM
;
STIME:      LD    DE,GETYEAR  ;Update years register
      LD    C,PRINT
      CALL  BDOS
      CALL  GETHEX
      JP    C,BEGINTIME
      PUSH  AF
      LD    A,09H       ;Years (Post 2000) register
      OUT   (RTCSEL),A
      POP   AF
      OUT   (RTCDATA),A ;Store in chip RAM (normally 1/100 & 1/10 sec RAM store)

      LD    A,05H       ;Mon, tue, wed... register
      OUT   (RTCSEL),A
      LD    DE,GETDOW
      LD    C,PRINT
      CALL  BDOS
      CALL  GETHEX
      JP    C,BEGINTIME
      OUT   (RTCDATA),A ;Store in chip register


      LD    A,04H       ;Hours register
      OUT   (RTCSEL),A
      LD    DE,GETHOUR  ;Get the hours value
      LD    C,PRINT
      CALL  BDOS
      CALL  GETHEX
      JP    C,BEGINTIME
      OUT   (RTCDATA),A
;
      LD    A,03H       ;Mins register
      OUT   (RTCSEL),A
      LD    DE,GETMINS  ;Get the mins value
      LD    C,PRINT
      CALL  BDOS
      CALL  GETHEX
      JP    C,BEGINTIME
      OUT   (RTCDATA),A
```

```
      LD    A,(CPM3_FLAG)     ;If CPM3 then skip setting year/day info
      OR    A,A
      JP    NZ,GOCMD    ;If CPM3 flag skip

      LD    A,07H       ;Month register
      OUT   (RTCSEL),A
      LD    DE,GETMONTH ;Get the Months value
      LD    C,PRINT
      CALL  BDOS
      CALL  GETHEX
      JP    C,BEGINTIME
      OUT   (RTCDATA),A

      LD    A,06H       ;Day of month register
      OUT   (RTCSEL),A
      LD    DE,GETDAY   ;Get the day value
      LD    C,PRINT
      CALL  BDOS
      CALL  GETHEX
      JP    C,BEGINTIME
      OUT   (RTCDATA),A
;
GOCMD:                        ;This command simply zeros seconds and lower fractions
      LD    A,15H       ;GO register. Sets sec and below to 0's
      OUT   (RTCSEL),A
      LD    A,0H
      OUT   (RTCDATA),A
;
      LD    DE,CRLFMSG
      LD    C,PRINT
      CALL  BDOS
      JP    GTIME       ;Display what we have
;
ZEROTIME:
      LD    DE,ZEROMSG
      LD    C,PRINT
      CALL  BDOS
      JP    GOCMD
;
;
;   <<<<<<<<<<< Just Dump the MM58167 Clock chip registers >>>>>>>>>>>>>>>>>>>>>>>>
;
DUMP_REGISTERS:
      LD    DE,DATA_DUMP_MSG
      LD    C,PRINT
      CALL  BDOS

      LD    DE,PORT_00000_MSG
      LD    C,PRINT
      CALL  BDOS
      LD    A,0H
      OUT   (RTCSEL),A  ;Point to milliseconds
      IN    A,(RTCDATA)
      CALL  PRINTREG
      CALL  ZBITS       ;Display bits

      LD    DE,PORT_00001_MSG
      LD    C,PRINT
      CALL  BDOS
      LD    A,01H
      OUT   (RTCSEL),A  ;Point to 1/00 and 1/10 seconds
      IN    A,(RTCDATA)
      CALL  PRINTREG
      CALL  ZBITS       ;Display bits

      LD    DE,PORT_00010_MSG
```

```
LD    C,PRINT
CALL  BDOS
LD    A,02H
OUT   (RTCSEL),A  ;Point to seconds
IN    A,(RTCDATA)
CALL  PRINTREG
CALL  ZBITS       ;Display bits

LD    DE,PORT_00011_MSG
LD    C,PRINT
CALL  BDOS
LD    A,03H
OUT   (RTCSEL),A  ;Point to minutes
IN    A,(RTCDATA)
CALL  PRINTREG
CALL  ZBITS       ;Display bits


LD    DE,PORT_00100_MSG
LD    C,PRINT
CALL  BDOS
LD    A,04H
OUT   (RTCSEL),A  ;Point to Hours
IN    A,(RTCDATA)
CALL  PRINTREG
CALL  ZBITS       ;Display bits

LD    DE,PORT_00101_MSG
LD    C,PRINT
CALL  BDOS
LD    A,05H
OUT   (RTCSEL),A  ;Point to DOW
IN    A,(RTCDATA)
CALL  PRINTREG
CALL  ZBITS       ;Display bits

LD    DE,PORT_00110_MSG
LD    C,PRINT
CALL  BDOS
LD    A,06H
OUT   (RTCSEL),A  ;Point to DOM
IN    A,(RTCDATA)
CALL  PRINTREG
CALL  ZBITS       ;Display bits

LD    DE,PORT_00111_MSG
LD    C,PRINT
CALL  BDOS
LD    A,07H
OUT   (RTCSEL),A  ;Point to Month
IN    A,(RTCDATA)
CALL  PRINTREG
CALL  ZBITS       ;Display bits

LD    DE,PORT_01000_MSG
LD    C,PRINT
CALL  BDOS
LD    A,08H
OUT   (RTCSEL),A  ;Point to RAM store of milliseconds
IN    A,(RTCDATA)
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits
```

```
LD    DE,PORT_01001_MSG
LD    C,PRINT
CALL  BDOS
LD    A,09H
OUT   (RTCSEL),A  ;Point to RAM store of 1/00 1nd 1/10 seconds
IN    A,(RTCDATA)
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_01010_MSG
LD    C,PRINT
CALL  BDOS
LD    A,0AH
OUT   (RTCSEL),A  ;Point to RAM store of Seconds
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_01011_MSG
LD    C,PRINT
CALL  BDOS
LD    A,0BH
OUT   (RTCSEL),A  ;Point to RAM store of Minutes
IN    A,(RTCDATA)
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_01100_MSG
LD    C,PRINT
CALL  BDOS
LD    A,0CH
OUT   (RTCSEL),A  ;Point to RAM store of Hours
IN    A,(RTCDATA)
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_01101_MSG
LD    C,PRINT
CALL  BDOS
LD    A,0DH
OUT   (RTCSEL),A  ;Point to RAM store of DOW
IN    A,(RTCDATA)
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_01110_MSG
```

```
LD    C,PRINT
CALL  BDOS
LD    A,0EH
OUT   (RTCSEL),A  ;Point to RAM store of DOM
IN    A,(RTCDATA)
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_01111_MSG
LD    C,PRINT
CALL  BDOS
LD    A,0FH
OUT   (RTCSEL),A  ;Point to RAM store of Months
IN    A,(RTCDATA)
CALL  LBYTE       ;Display actual Hex data (not BCD)
LD    DE,HEX_MSG
LD    C,PRINT
CALL  BDOS
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits


LD    DE,MORE_MSG ;Enter any key to continue...
LD    C,PRINT
CALL  BDOS
LD    C,RDCON           ;CP/M Read character
CALL  BDOS


LD    DE,PORT_10000_MSG
LD    C,PRINT
CALL  BDOS
LD    A,10H
OUT   (RTCSEL),A  ;Interrupt Status register
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_10001_MSG
LD    C,PRINT
CALL  BDOS
LD    A,11H
OUT   (RTCSEL),A  ;Interrupt Control register
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_10010_MSG
LD    C,PRINT
CALL  BDOS
LD    A,12H
OUT   (RTCSEL),A  ;Counter reset register
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_10011_MSG
LD    C,PRINT
CALL  BDOS
LD    A,13H
OUT   (RTCSEL),A  ;RAM reset register
IN    A,(RTCDATA)
CALL  ZBITS       ;Display bits

LD    DE,PORT_10100_MSG
```

```
      LD    C,PRINT
      CALL  BDOS
      LD    A,14H
      OUT   (RTCSEL),A  ;Status bit
      IN    A,(RTCDATA)
      CALL  ZBITS       ;Display bits

      LD    DE,PORT_10101_MSG
      LD    C,PRINT
      CALL  BDOS
      LD    A,15H
      OUT   (RTCSEL),A  ;GO Command
      IN    A,(RTCDATA)
      CALL  ZBITS       ;Display bits

      LD    DE,PORT_10110_MSG
      LD    C,PRINT
      CALL  BDOS
      LD    A,16H
      OUT   (RTCSEL),A  ;Standby Interrypt
      IN    A,(RTCDATA)
      CALL  ZBITS       ;Display bits

      LD    DE,PORT_11111_MSG
      LD    C,PRINT
      CALL  BDOS
      LD    A,16H
      OUT   (RTCSEL),A  ;Test Mode
      IN    A,(RTCDATA)
      CALL  ZBITS       ;Display bits


      LD    DE,CRLFMSG
      LD    C,PRINT
      CALL  BDOS
      JP    BEGINTIME   ;We are done
;
;
INT_LOOP:
      DI                 ;Prevent interrupts for now (BIOS may not be setup correctly)
      LD    DE,SET_INTERRUPT ;Test the ability of the clock chip to trigger interrupts
      LD    C,PRINT
      CALL  BDOS

      LD    A,11H       ;Point to Interrupt Control Register
      OUT   (RTCSEL),A
      LD    A,00000100B ;Set 1 Second interrupt bit
      OUT   (RTCDATA),A

      LD    A,16H       ;Point to Standby Interrupt Register (Pin #14)
      OUT   (RTCSEL),A
      LD    A,00000001B ;Turn it on
      OUT   (RTCDATA),A

ILOOP:      LD    B,0FFH            ;Delay a little
MM2:  PUSH  BC
      LD    E,'.'
      LD    C,WRCON           ;Write to consol
      CALL  BDOS
      POP   BC
      DEC   B
      JP    NZ,MM2

      CALL  ZCRLF
      LD    A,10H       ;Point to Interrupt Status Register
      OUT   (RTCSEL),A
```

```
      IN    A,(RTCDATA)
      CALL  ZBITS       ;Display bits continously. One bit should appear every second!
      CALL  ZCRLF

      LD    C,CONSTAT   ;Is there a keyboard abort request (ESC)
      CALL  BDOS
      OR    A,A
      JP    Z,ILOOP           ;Z if nothing
      LD    C,RDCON           ;If so is it an ESC character
      CALL  BDOS
      CP    A,ESC
      JP    NZ,ILOOP    ;Note pin #13 of chip will generate an interrupt
      JP    BEGINTIME   ;Abort with ESC key
;
;
;----------------------------- SUPPORT ROUTINES ----------------------------------
;
;Convert a CPM3 day count in [HL] to [D]=day [E]=month [HL]=year (all in Hex)
;
C3DMY:      PUSH  IY          ;Will need these below
      PUSH  IX
      PUSH  BC
                        ;HL = days since 2 Jan 1976 (so 1 Jan is Day 0)
                        ;This makes month days come out right at the end
      LD    DE,730            ;(2DAH) adjust to 1/1/1978 to make leap year
calculations easier.
      add   hl,de
                        ;Divide by 1461 to get number of 4-year cycles since 1 Jan
1976
      PUSH  HL          ;Save Days since 1/1/1976
      ld    de,1461
      call  DIVHD
      ld    a,l         ;A = 4-year cycles since 1/1/1976. The CP/M clock
      add   a,a         ;X4
      add   a,a         ;A = no. of years since 1/1/1976
      call  MULHD       ;HL X 1461 = no. of days in the cycles
      and   a
      POP   de          ;HL -> DE = days since 1/1/1976
      ex    de,hl
      sbc   hl,de       ;HL = days since start of cycle
                        ;If daycount is 59, this is 29 Feb.
      ld    c,a         ;C = years since 1/1/1976
      ld    a,h
      or    a
      jr    nz,notf29
      ld    a,l
      cp    59
      jr    z,f29
      jr    c,ltf29
notf29:     dec   hl          ;HL = days since start of cycle, taking account
                        ;of leap years.
ltf29:      PUSH  HL
      ld    de,365
      call  DIVHD       ;L = years since start of cycle

      ld    a,l
      add   a,c         ;A = years since 1/1/1976
      ld    c,a         ;C = years since 1/1/1976
      PUSH  BC
      POP   IY          ;Temporarly use IY to store BC

      call  MULHD       ;HL X DE = days to start of year
      POP   de          ;DE (from HL push above) has days since start of cycle
      ex    de,hl
      and   a
      sbc   hl,de       ;HL = days since start of year
```

```
      ld    b,0         ;B = month
      ld    ix,c3mdys
      ld    (ix+1),28
mclp: inc   b
      ld    e,(ix+0)
      inc   ix
      ld    d,0         ;DE = days in this month
      and   a
      sbc   hl,de
      jr    nc,mclp
      add   hl,de       ;HL = day of month, B = month, C = years since 1976
      ld    d,l
      inc   d           ;Make the days 1-based
      ld    e,b
      jr    fret
;
f29:  ld    de,1D02h    ;= 29 Feb
fret: ld    b,0         ;BC = years since 1/1/1976
      ld    hl,1976
      PUSH  IY          ;Get back old BC value
      POP   BC
      add   hl,bc       ;return with D=day, E=month, HL=year (all in HEX)
      POP   BC          ;restore these
      POP   IX
      POP   IY
      ret
;
c3mdys:     db    31,28,31,30,31,30,31,31,30,31,30,31
;
;
;MULHD -- MULTIPLY HL BY DE, RESULT IN HL CARRY SET MEANS OVERFLOW
;         Note [A] IS DESTROYED
;
MULHD:      PUSH  AF
      PUSH  BC          ; SAVE REGS
      PUSH  DE
      XOR   A           ; SET NO OVFL
      LD    (OVFL),A
      PUSH  HL          ; SAVE HL
      LD    HL,0        ; ZERO LONG ACC
      LD    (ACC),HL
      POP   HL          ; GET HL
      LD    B,16        ; 16 LOOPS
MLHD: CALL  SHFTRHX           ; SHIFT RIGHT MULTIPLIER
      JP    NC,MLHD1    ; DON'T ADD IN IF LSB IS ZERO
      PUSH  HL          ; SAVE VALUE
      LD    HL,(ACC)
      LD    A,L         ; ADD IN MULTIPLICAND
      ADD   A,E
      LD    L,A
      LD    A,H
      ADC   A,D
      LD    H,A
      LD    (ACC),HL    ; NEW ACCUMULATED VALUE
      JP    NC,MLHD0
      LD    A,0FFH            ; SET OVFL FLAG
      LD    (OVFL),A
MLHD0:      POP   HL          ; GET VALUE
MLHD1:      EX    DE,HL       ; SHIFT LEFT MULTIPLICAND
      CALL  SHFTLHX
      EX    DE,HL
      DEC   B           ; COUNT DOWN
      JP    NZ,MLHD
      POP   DE          ; RESTORE REGS
      POP   BC
      LD    HL,(ACC)    ; GET RESULT
```

```
      LD    A,(OVFL)    ; OVERFLOW?
      OR    A           ; 0=NO
      JP    Z,OKRET
OVFLRET:POP AF          ; GET PSW
      SCF               ; SET CARRY
      RET
OKRET:      POP   AF
      OR    A           ; CLEAR CARRY
      RET
;
;
;  SHIFT ROUTINES FOR MULHD
;
SHFTRHX:                ; SHIFT RIGHT
      PUSH  AF
      AND   A           ; CLEAR CARRY
      LD    A,H         ; SHIFT HIGH
      RRA               ; ROTATE 9-BIT ACC RIGHT
      LD    H,A
      LD    A,L         ; SHIFT LOW
      RRA
      LD    L,A
      JP    NC,OKRET
      JP    OVFLRET
SHFTLHX:                ; SHIFT LEFT
      PUSH  AF
      AND   A           ; CLEAR CARRY
      LD    A,L         ; SHIFT LOW
      RLA               ; ROTATE 9-BIT ACC LEFT
      LD    L,A
      LD    A,H         ; SHIFT HIGH
      RLA
      LD    H,A
      JP    NC,OKRET
      JP    OVFLRET
;
;
;DIVHD -- DIVIDE HL BY DE, RESULT IN HL
;
DIVHD:
      PUSH  AF
      PUSH  BC
      PUSH  HL          ; SAVE HL
      LD    HL,0
      LD    (ACC),HL    ; ZERO ACCUMULATOR
      POP   HL
                        ; CHECK TO MAKE SURE THAT HL > DE
      LD    A,H
      CP    D           ; H > D?
      JP    C,DOVFL           ; ZERO RESULT IF H < D
      JP    NZ,DNOVFL
      LD    A,L
      CP    E           ; L > E?
      JP    C,DOVFL           ; ZERO RESULT IF HL < DE
DNOVFL:     LD    B,16        ; 16 LOOPS
DVHD: CALL  SHFTLH            ; SHIFT DIVIDEND LEFT
      PUSH  HL          ; SAVE DIVIDEND
      LD    HL,(ACC)    ; GET ACC
      CALL  SHFLCH            ; ROTATE ACC AND MOVE IN CARRY
      LD    (ACC),HL    ; NEW ACC
      LD    A,L         ; LOW COMPARISON
      SUB   E           ; COMPARE AND SUBTRACT
      LD    L,A
      LD    A,H         ; HIGH COMPARISON
      SBC   A,D         ; COMPARE AND SUBTRACT
      LD    H,A
```

```
      JP    C,DVHD0
      LD    (ACC),HL    ; SAVE NEW ACC
      POP   HL          ; GET DIVIDEND
      LD    A,L         ; PLACE IN A 1 TO QUOTIENT
      OR    1
      LD    L,A
      JP    DVHD1
DVHD0:      POP   HL          ; GET DIVIDEND AND LEAVE LSB AT ZERO
DVHD1:      DEC   B           ; COUNT DOWN
      JP    NZ,DVHD
DVRET:      POP   BC
      POP   AF
      RET
DOVFL:      LD    HL,0        ; RESULT IS ZERO
      JP    DVRET

;
;  SHIFT ROUTINES FOR DIVHD
;
SHFTLH:                       ; SHIFT HL LEFT
      PUSH  AF
      AND   A           ; CLEAR CARRY
SHFL: LD    A,L         ; SHIFT LOW
      RLA               ; ROTATE 9-BIT ACC LEFT
      LD    L,A
      LD    A,H         ; SHIFT HIGH
      RLA
      LD    H,A
      JP    NC,OKRETX
      POP   AF
      SCF               ; SET CARRY FOR OVERFLOW
      RET
OKRETX:     POP   AF
      OR    A           ; CLEAR CARRY FOR NO OVERFLOW
      RET
SHFLCH:                       ; SHIFT HL LEFT, BUT SHIFT IN CARRY FLAG
      PUSH  AF
      JP    SHFL
;
;  BUFFERS
;
ACC:  DS    2           ; TEMP ACCUMULATOR
OVFL: DS    1           ; OVERFLOW FLAG
;
; Print BCD values in chips registers
;
PRINTREG:               ;Display Clock register as High/Low byte on CRT
      PUSH  AF          ;return with same value in A
      PUSH  AF          ;High & low minutes
      RRA
      RRA
      RRA
      RRA
      AND   0FH
      ADD   A,30H
      LD    E,A
      LD    C,WRCON           ;Write high byte mins
      CALL  BDOS

      POP   AF
      AND   0FH
      ADD   A,30H
      LD    E,A
      LD    C,WRCON           ;Write low byte mins
      CALL  BDOS
      POP   AF
```

```
      RET
;
; Print Hex value of [A]
;
LBYTE:
      PUSH  AF
      RRCA
      RRCA
      RRCA
      RRCA
      CALL  SF598
      POP   AF
SF598:
      CALL  CONV
      JP    ZCO         ;Send to consol
;
; CONVERT HEX TO ASCII
;
CONV:
      AND   0FH
      ADD   A,90H
      DAA
      ADC   A,40H
      DAA
      LD    C,A
      RET
;
; Main consol output routine using CPM. Character in [C]. All registers unchanged
;
ZCO:
      PUSH  AF
      PUSH  BC
      PUSH  DE
      LD    E,C
      LD    C,WRCON
      CALL  BDOS
      POP   DE
      POP   BC
      POP   AF
      RET
;
; Send CR/LF to Consol
;
ZCRLF:      LD    C,CR
      CALL  ZCO
      LD    C,LF
      CALL  ZCO
      RET
;
;
;Return with 2 digits in [A]. If abort, Carry flag set + ESC in [A]
;
GETHEX:
      PUSH  BC
      LD    C,RDCON
      CALL  BDOS        ;Get a character from keyboard & ECHO
      CP    A,ESC
      JR    Z,HEXABORT
      CP    '/'         ;check 0-9, A-F
      JR    C,HEXABORT
      CP    '9'+1
      JR    NC,HEXABORT
      CALL  ASBIN       ;Convert to binary
      SLA   A
      SLA   A
      SLA   A
```

```
      SLA   A           ;Shift to high nibble
      LD    B,A         ;Store it
      PUSH  BC          ;Because CP/M destroys BC
      LD    C,RDCON
      CALL  BDOS        ;Get a character from keyboard & ECHO
      POP   BC
      CP    A,ESC
      JR    Z,HEXABORT
      CP    '/'         ;check 0-9, A-F
      JR    C,HEXABORT
      CP    '9'+1
      JR    NC,HEXABORT
      CALL  ASBIN       ;Convert to binary
      OR    A,B         ;add in the first digit
      OR    A,A         ;To return NC
      POP   BC
      RET
;
HEXABORT:
      SCF               ;Set Carry flag
      LD    A,ESC
      POP   BC
      RET
;
; ASCII TO BINARY CONVERSION ROUTINE
;
ASBIN:
      SUB   30H
      CP    0AH
      RET   M
      SUB   07H
      RET
;
; DISPLAY BIT PATTERN IN [A]
;
ZBITS:
      PUSH  AF
      PUSH  BC
      PUSH  DE
      PUSH  AF
      LD    C,' '
      CALL  ZCO
      LD    C,'('
      CALL  ZCO
      POP   AF
      LD    E,A
      LD    B,8
BQ2:  SLA   E
      LD    A,18H
      ADC   A,A
      LD    C,A
      CALL  ZCO
      DJNZ  BQ2

      LD    C,')'
      CALL  ZCO
      POP   DE
      POP   BC
      POP   AF
      RET
;
;
; Binary to ASCII Decimal String Conversion Print Routine
; [DE] contains binary number to be displayed. String is built right->left in buffer
; Then printed out via CPM.
;
```

```
PRINT_DEC:
      LD    HL,BUFFER   ;Location to hold string
      LD    (BUFPTR),HL ;Save buffer pointer
      EX    DE,HL
      LD    A,0
      LD    (CURLEN),A  ;Current length of buffer is 0
      LD    A,H
      LD    (NGFLAG),A  ;Save Sign Value
      OR    A           ;Set FLAGS Value
      JP    P,CNVERT    ;JMP IF VALYE IS +
      EX    DE,HL
      LD    HL,0
      OR    A           ;CLEAR Carry
      SBC   HL,DE
                        ;Convert Value to a String
CNVERT:
      LD    E,0         ;HL := HL DIV 10,  DE := HL MOD 10

      LD    B,16
      OR    A
DVLOOP:
      RL    L           ;Do a 24-bit shift left
      RL    H
      RL    E
      LD    A,E
      SUB   10          ;If remainder is 10 or more next 1
      CCF
      JR    NC,DECCNT
      LD    E,A
DECCNT:
      DJNZ  DVLOOP            ;continue until all bits are done
      RL    L
      RL    H
CHINS:
      LD    A,E         ;Insert next character into buffer
      ADD   A,'0'
      CALL  INSERT
      LD    A,H
      OR    L
      JR    NZ,CNVERT

EXIT: LD    A,(NGFLAG)  ;All done
      OR    A           ;Check it + or - overall
      JP    P,POS
      LD    A,'-'
      CALL  INSERT
POS:  LD    HL,BUFFER
      LD    A,(HL)            ;number of characters in buffer
      INC   A
      LD    C,A
      LD    B,0
      ADD   HL,BC       ;Point to end of buffer
      LD    (HL),'$'    ;Put in terminating character for CPM
      LD    DE,BUFFER+1 ;Now print the complete string (+1, because buffer count is in
first posttion)
      LD    C,PRINT
      CALL  BDOS
      RET

INSERT:
      PUSH  HL          ;Save for now
      PUSH  AF
      LD    HL,(BUFPTR) ;Current end of buffer
      LD    D,H
      LD    E,L
      INC   DE          ;DE = End+1
```

```
      LD    (BUFPTR),DE
      LD    A,(CURLEN)
      OR    A
      JR    Z,EXITMR
      LD    C,A
      LD    B,0
      LDDR              ;Move entire buffer up one byte
EXITMR:
      LD    A,(CURLEN)
      INC   A
      LD    (CURLEN),A
      LD    (HL),A
      EX    DE,HL
      POP   AF
      LD    (HL),A
      POP   HL
      RET

BUFPTR:     DS    2           ;pointer to last character in buffer
CURLEN:     DS    1           ;Current length
NGFLAG:     DS    1
BUFFER:     DB    '           $'    ;Decimal string will be built here

CPM3_FLAG: DB     0FFH        ;Flag used by program for CPM3 register protection

;-------------------------------------------------------------------------------
--------------

SIGNON:           DB    CR,LF,LF
            DB    'Program to Configure the S-100 PIC/RTC MM58167A Clock-Calander
Board.'
            DB    CR,LF,'Please note certain Registers/RAM addersses are reserved for
the CPM3'
            DB    CR,LF,'clock device driver to store data when power is off.'
            DB    CR,LF,'They will not be altered in this program by default.'
            DB    CR,LF,'MM58167A Register Select Port = $'
SIGNON2:    DB    'H.   MM58167A Data Port = $'
SIGNON3:    DB    'H.',CR,LF,'$'
CRLFMSG:    DB    CR,LF,'$'
TIMEMENU:   DB    CR,LF
            DB    'Configure MM58167A Clock Chip.',CR,LF
            DB    '0 = Show Time & Date',CR,LF
            DB    '1 = Get Time Information',CR,LF
            DB    '2 = Set Clock/Time',CR,LF
            DB    '3 = Zero seconds',CR,LF
            DB    '4 = Raw MM58167A Registers & RAM data dump',CR,LF
            DB    '5 = Do 1 Second Interrupt Test',CR,LF,'$'

TIMEMENU1:  DB    '6 = Turn ON protection of CPM3 utilized Registers/RAM',CR,LF
            DB    'ESC to Skip',CR,LF
            DB    'Menu choice ---->$'
TIMEMENU2:  DB    '6 = Toggle OFF protection of CPM3 utilized Registers/RAM',CR,LF
            DB    'ESC to Skip',CR,LF
            DB    'Menu choice ---->$'

CPM3_ON_MSG:      DB    'Registers/RAM used by CPM3 is Protected.',CR,LF,'$'
CPM3_OFF_MSG      DB    'Registers/RAM used by CPM3 is NOT Protected and will be
displayed here.',CR,LF,'$'

ABORTMSG:   DB    CR,LF
            DB    'Invalid menu item',CR,LF,'$'
TIMEMSG:    DB    CR,LF,'Time: $'
YEAR2000:   DB    '/20$'
GETMONTH:   DB    CR,LF
            DB    'Enter Month, two characters, (01-12)--->$'
GETDAY:           DB    CR,LF
```

```
            DB    'Enter Day, two characters, (01-31)--->$'
GETYEAR:    DB    CR,LF
            DB    'Enter Year after 2000, two characters, (01-99)--->$'
GETDOW:             DB    CR,LF
            DB    'Enter day of week, (01=Mon,02=Tue....07=Sun)--->$'
GETHOUR:    DB    CR,LF
            DB    'Enter Hour, two characters, (00-23)--->$'
GETMINS:    DB    CR,LF
            DB    'Enter Minutes, two characters, (00-59)--->$'
ZEROMSG             DB    CR,LF,'Reset clock seconds.',CR,LF,'$'
SET_INTERRUPT       DB    CR,LF,'1 Second Interrupt Test.'CR,LF,'$'
;
;
TIME_MSG:   DB    CR,LF,'Time = $'
DATE_MSG:   DB    '   Date = $'
YEAR_MSG    DB    '/20$'

DATA_MSG:   DB    CR,LF,'Chip Registers Data:-',CR,LF,'$'
DATA_MSG3:  DB    CR,LF,'Chip Registers Data (CPM3 is utilizing Date Registers/RAM):-
',CR,LF,'$'
PORT_SECS_MSG       DB    ' <--Seconds Reg',CR,LF,'$'
PORT_MINS_MSG       DB    ' <--Min Reg   $'
PORT_HRS_MSG        DB    ' <--Hours Reg',CR,LF,'$'
PORT_DOW_MSG        DB    ' <--Day of Week Reg   $'
PORT_DOM_MSG        DB    ' <--Day of Month Reg   $'
PORT_MONTH_MSG      DB    ' <--Month Reg',CR,LF,'$'

RAM_YEAR_STORE_MSG DB   ' <--Year Store RAM',CR,LF,'$'

RAM_MINS_MSG3       DB    'H <--Min RAM (Used by CPM3 to store Low Byte of 16 bit days
count)',CR,LF,'$'
RAM_HRS_MSG3        DB    'H <--Hours RAM  (Used by CPM3 to store High Byte of 16 bit
days count)',CR,LF,'$'
RAM_MINS_MSG        DB    'H <--Min RAM',CR,LF,'$'
RAM_HRS_MSG DB    'H <--Hours RAM',CR,LF,'$'

RAM_DOW_MSG DB    ' <--Day of Week RAM',CR,LF,'$'

RAM_DOM_MSG3        DB    ' <--"Day of Month RAM" (CPM, days since last
reboot)'CR,LF,'$'
RAM_MONTH_MSG3      DB    ' <--"Month RAM" (CPM, months since last reboot)',CR,LF,'$'
RAM_DOM_MSG DB    ' <--Day of Month RAM'CR,LF,'$'
RAM_MONTH_MSG       DB    ' <--Month RAM',CR,LF,'$'
CPM_DAYS_MSG        DB    'H <--CPM3 Store of days since 1/1/1978',CR,LF,'$'
CPM_DAYMSG  DB    'H <--CPM Day  $'
CPM_MONTHMSG        DB    'H <--CPM Month $'
CPM_YEARMSG DB    'H <--CPM Year$'
CPM_CALCULATED_DATE_MSG  DB   'Calculated CPM Date ($'
BRACKET_MSG DB    ') = $'

;
DATA_DUMP_MSG       DB    CR,LF,'Dump of all the MM58167A registers:-$'
PORT_00000_MSG      DB    CR,LF,'Address 00000 (Milli-Seconds) = $'
PORT_00001_MSG      DB    CR,LF,'Address 00001 (1/100 & 1/10 Seconds) = $'
PORT_00010_MSG      DB    CR,LF,'Address 00010 (Seconds) = $'
PORT_00011_MSG      DB    CR,LF,'Address 00011 (Minutes) = $'
PORT_00100_MSG      DB    CR,LF,'Address 00100 (Hours) = $'
PORT_00101_MSG      DB    CR,LF,'Address 00101 (Day of Week) = $'
PORT_00110_MSG      DB    CR,LF,'Address 00110 (Day of Month) = $'
PORT_00111_MSG      DB    CR,LF,'Address 00111 (Month) = $'

PORT_01000_MSG      DB    CR,LF,LF,'Address 01000 (RAM store Milliseconds) = $'
PORT_01001_MSG      DB    CR,LF,'Address 01001 (RAM store 1/100 & 1/10 Seconds) = $'
PORT_01010_MSG      DB    CR,LF,'Address 01010 (RAM store Seconds) = $'
PORT_01011_MSG      DB    CR,LF,'Address 01011 (RAM store Minutes) = $'
PORT_01100_MSG      DB    CR,LF,'Address 01100 (RAM store Hours) = $'
```

```
PORT_01101_MSG    DB    CR,LF,'Address 01101 (RAM store Day of Week) = $'
PORT_01110_MSG    DB    CR,LF,'Address 01110 (RAM store Day of Month) = $'
PORT_01111_MSG    DB    CR,LF,'Address 01111 (RAM store Month) = $'
MORE_MSG    DB    CR,LF,'Enter any key to continue....$'
HEX_MSG           DB    'hex. $'

PORT_10000_MSG    DB    CR,LF,'Address 10000 (Interrupt Status Register)= $'
PORT_10001_MSG    DB    CR,LF,'Address 10001 (Interrupt Control Register)= $'
PORT_10010_MSG    DB    CR,LF,'Address 10010 (Counters Reset)= $'
PORT_10011_MSG    DB    CR,LF,'Address 10011 (RAM reset)= $'
PORT_10100_MSG    DB    CR,LF,'Address 10100 (Status bit)= $'
PORT_10101_MSG    DB    CR,LF,'Address 10101 (GO command)= $'
PORT_10110_MSG    DB    CR,LF,'Address 10110 (Standby Interrupt*)= $'
PORT_11111_MSG    DB    CR,LF,'Address 11111 (Test Mode)= $'
;
      DS    40H
STACK:      DB    0H
;
; END
```