```
#pragma config OSC = HS
#pragma config FCMEN = OFF
#pragma config IESO = OFF

#pragma config PWRT = OFF
#pragma config BOREN = SBORDIS
#pragma config BORV = 3

#pragma config WDT = OFF
#pragma config WDTPS = 32768

#pragma config CCP2MX = PORTC
#pragma config PBADEN = OFF
#pragma config LPT1OSC = OFF
#pragma config MCLRE = ON

#pragma config STVREN = OFF
#pragma config LVP = OFF
#pragma config XINST = OFF


#pragma config CP0 = OFF
#pragma config CP1 = OFF
#pragma config CP2 = OFF
#pragma config CP3 = OFF


#pragma config CPB = OFF
#pragma config CPD = OFF
```

```
#pragma config WRT0 = OFF

#pragma config WRT1 = OFF

#pragma config WRT2 = OFF

#pragma config WRT3 = OFF



#pragma config WRTC = OFF

#pragma config WRTB = OFF

#pragma config WRTD = OFF



#pragma config EBTR0 = OFF

#pragma config EBTR1 = OFF

#pragma config EBTR2 = OFF

#pragma config EBTR3 = OFF



#pragma config EBTRB = OFF



#include <xc.h>

#include <pic18f4520.h>


#define _XTAL_FREQ 20000000



#define RS LATBbits.LATB2

#define EN LATBbits.LATB3

#define D4 LATBbits.LATB4

#define D5 LATBbits.LATB5
```

```
#define D6 LATBbits.LATB6

#define D7 LATBbits.LATB7



#define ALMtd 200


short bz;


void Lcd_Port(char a)

{

        if(a & 1)

                D4 = 1;

        else

                D4 = 0;


        if(a & 2)

                D5 = 1;

        else

                D5 = 0;


        if(a & 4)

                D6 = 1;

        else

                D6 = 0;


        if(a & 8)

                D7 = 1;

        else

                D7 = 0;

}

void Lcd_Cmd(char a)
```

```
{
            RS = 0;
            Lcd_Port(a);
            EN  = 1;
   __delay_ms(1);
   EN  = 0;
}

void Lcd_Clear()
{
            Lcd_Cmd(0);
            Lcd_Cmd(1);
   __delay_ms(1);
}

void Lcd_Cursor_ON()
{
            Lcd_Cmd(0);
            Lcd_Cmd(0xF);
   __delay_ms(1);
}

void Lcd_Cursor_OFF()
{
            Lcd_Cmd(0);
            Lcd_Cmd(0xC);
   __delay_ms(1);
}

void Lcd_Set_Cursor(char a, char b)
{
```

```
              char temp,z,y;
              if(a == 1)
              {
                temp = 0x80 + b - 1;
                            z = temp>>4;
                            y = temp & 0x0F;
                            Lcd_Cmd(z);
                            Lcd_Cmd(y);
              }
              else if(a == 2)
              {
                            temp = 0xC0 + b - 1;
                            z = temp>>4;
                            y = temp & 0x0F;
                            Lcd_Cmd(z);
                            Lcd_Cmd(y);
              }
}

void Lcd_Init()
{
    Lcd_Port(0x00);
    __delay_ms(20);
    Lcd_Cmd(0x03);
              __delay_ms(5);
    Lcd_Cmd(0x03);
               __delay_ms(11);
    Lcd_Cmd(0x03);
    /////////////////////////////////////////////////////
    Lcd_Cmd(0x02);
    Lcd_Cmd(0x02);
```

```
   Lcd_Cmd(0x08);
   Lcd_Cmd(0x00);
   Lcd_Cmd(0x0C);
   Lcd_Cmd(0x00);
   Lcd_Cmd(0x06);
}

void Lcd_Write_Char(char a)
{
   char temp,y;
   temp = a&0x0F;
   y = a&0xF0;
   RS = 1;
   Lcd_Port(y>>4);
   EN = 1;
   __delay_us(40);
   EN = 0;
   Lcd_Port(temp);
   EN = 1;
   __delay_us(40);
   EN = 0;
   __delay_ms(1);
}

void Lcd_Write_String(char *a)
{
          int i;
          for(i=0;a[i]!='\0';i++)
            Lcd_Write_Char(a[i]);
}
```

```
void spiMasterInit( void )
{
   SSPSTAT = 0x00;
   SSPCON1 = 0x22;

   TRISCbits.RC3 = 0;
   TRISCbits.RC5 = 0;
}

void writeSPI(char dat)
{
   SSPBUF = dat;
   __delay_ms(1);
}

short spiDataReady()
{
   if(SSPSTATbits.BF)
      return 1;
   else
      return 0;
}

char readSPI()
{
   while ( !SSPSTATbits.BF );
   return(SSPBUF);
}

unsigned short readDS3234(unsigned short address)
{
   PORTBbits.RB1 = 0;
   __delay_us(10);
   writeSPI(0x00 + address);
   writeSPI(0x00 + address);
   __delay_us(10);
   PORTBbits.RB1 = 1;
   return(readSPI());
}

void writeDS3234(unsigned short address, unsigned short data)
{
   PORTBbits.RB1 = 0;
   __delay_us(10);
   writeSPI(0x80 + address);
   writeSPI(data);
   __delay_us(10);
   PORTBbits.RB1 = 1;
}

unsigned char BCD2UpperCh(unsigned short bcd)
{
   return ((bcd >> 4) + '0');
}

unsigned char BCD2LowerCh(unsigned short bcd)
{
   return ((bcd & 0x0F) + '0');
}

unsigned short Binary2BCD(unsigned short a)
{
   int t1, t2;
   t1 = a%10;
   t1 = t1 & 0x0F;
   a = a/10;
   t2 = a%10;
   t2 = 0x0F & t2;
   t2 = t2 << 4;
   t2 = 0xF0 & t2;
   t1 = t1 | t2;
   return t1;
}

unsigned short BCD2Binary(unsigned short a)
{
   int r,t;
   t = a & 0x0F;
   r = t;
   a = 0xF0 & a;
   t = a >> 4;
   t = 0x0F & t;
   r = t*10 + r;
   return r;
}

short readKeypad()
{
   LATDbits.LATD3 = 1;
   __delay_us(10);
   if(PORTDbits.RD0 == 1)
   {
       while(PORTDbits.RD0 == 1);
       return 3;
   }
   else if(PORTDbits.RD1 == 1)
   {
       while(PORTDbits.RD1 == 1);
       return 2;
   }
   else if(PORTDbits.RD2 == 1)
   {
       while(PORTDbits.RD2 == 1);
       return 1;
   }

   LATDbits.LATD3 = 0;
   LATDbits.LATD4 = 1;
   __delay_us(10);

   if(PORTDbits.RD0 == 1)
   {
       while(PORTDbits.RD0 == 1);
       return 6;
   }
   else if(PORTDbits.RD1 == 1)
   {
       while(PORTDbits.RD1 == 1);
       return 5;
   }
   else if(PORTDbits.RD2 == 1)
   {
       while(PORTDbits.RD2 == 1);
       return 4;
```

```
}

LATDbits.LATD4 = 0;
LATDbits.LATD5 = 1;
__delay_us(10);

if(PORTDbits.RD0 == 1)
{
   while(PORTDbits.RD0 == 1);
   return 9;
}
else if(PORTDbits.RD1 == 1)
{
   while(PORTDbits.RD1 == 1);
   return 8;
}
else if(PORTDbits.RD2 == 1)
{
   while(PORTDbits.RD2 == 1);
   return 7;
}

LATDbits.LATD5 = 0;
LATDbits.LATD6 = 1;
__delay_us(10);

if(PORTDbits.RD0 == 1)
{
   while(PORTDbits.RD0 == 1);
   return 12;
}
```

```
    else if(PORTDbits.RD1 == 1)
    {
        while(PORTDbits.RD1 == 1);
        return 11;
    }
    else if(PORTDbits.RD2 == 1)
    {
        while(PORTDbits.RD2 == 1);
        return 10;
    }
    LATDbits.LATD6 = 0;
    return 0;
}

char decodeCharacter(short a)
{
    switch(a)
    {
        case 11 : return '0';
        case 1 : return '1';
        case 2 : return '2';
        case 3 : return '3';
        case 4 : return '4';
        case 5 : return '5';
        case 6 : return '6';
        case 7 : return '7';
        case 8 : return '8';
        case 9 : return '9';
    }
}
```

```
__interrupt() void alarm(void)
{
   if(INT0IF == 1)
   {
      bz = 3;
      writeDS3234(0x0F,0xC8);
      INT0IF = 0;
   }
}

void main(void)
{
   unsigned short second, minute, hour, hr, day, dday, month, year, ap, m;
   unsigned short aSecond,aMinute,aHour,aHr,aAP,aTM1,aTM2;
   short kp,mt,temp;
   char t;
   char time[] = "00:00:00 PM";
   char aTime[] = "00:00:00   ";
   char date[] = "00-00-00   ";



   TRISD = 0x07;
   LATD = 0x00;
   TRISB = 0x01;
   GIE = 1;
   PEIE = 1;
   INT0IF = 0;
   INTEDG0 = 0;
   INT0IE = 1;

   PORTBbits.RB1 = 1;
```

```
m = 0;
mt = 0;
bz = 0;
spiMasterInit();
Lcd_Init();
Lcd_Clear();
aTM1 = 0;
aTM2 = 0;

while(1)
{
  second = readDS3234(0);
  minute = readDS3234(1);
  hour = readDS3234(2);
  hr = hour & 0b00011111;
  ap = hour & 0b00100000;
  dday = readDS3234(3);
  day = readDS3234(4);
  month = readDS3234(5);
  year = readDS3234(6);

  time[0] = BCD2UpperCh(hr);
  time[1] = BCD2LowerCh(hr);
  time[3] = BCD2UpperCh(minute);
  time[4] = BCD2LowerCh(minute);
  time[6] = BCD2UpperCh(second);
  time[7] = BCD2LowerCh(second);

  if(bz)
  {
    if(aTM1 < ALMtd)
    {
        LATDbits.LATD7 = 1;
        aTM1++;
    }
    else if(aTM1 >= ALMtd && aTM2 < ALMtd)
    {
        LATDbits.LATD7 = 0;
        aTM2++;
    }
    else
    {
        aTM1 = 0;
        aTM2 = 0;
        bz--;
    }
}
else
   LATDbits.LATD7 = 0;

if(ap)
{
    time[9] = 'P';
    time[10] = 'M';
}
else
{
    time[9] = 'A';
    time[10] = 'M';
}

date[0] = BCD2UpperCh(day);
```

```
date[1] = BCD2LowerCh(day);
date[3] = BCD2UpperCh(month);
date[4] = BCD2LowerCh(month);
date[6] = BCD2UpperCh(year);
date[7] = BCD2LowerCh(year);

kp = readKeypad();
t = 0;
if(kp == 10)
{
   m++;
   mt = 0;
   if(m > 3)
      m = 0;
}
else if((kp > 0 && kp < 10) || kp == 11)
{
   if(m)
   {
      t = decodeCharacter(kp);
      mt++;
   }
}
else if(kp == 12)
{
   m = 0;
   bz = 0;
}

if(m == 0)
{
   Lcd_Set_Cursor(1,1);
   Lcd_Write_String("Time: ");
   Lcd_Write_String(time);
   Lcd_Set_Cursor(2,1);
   Lcd_Write_String("Date: ");
   Lcd_Write_String(date);
}
else if(m == 1)
{
   aSecond = readDS3234(0x07);
   aMinute = readDS3234(0x08);
   aHour = readDS3234(0x09);
   aHr = aHour & 0b00011111;
   aAP = aHour & 0b00100000;
   aTime[0] = BCD2UpperCh(aHr);
   aTime[1] = BCD2LowerCh(aHr);
   aTime[3] = BCD2UpperCh(aMinute);
   aTime[4] = BCD2LowerCh(aMinute);
   aTime[6] = BCD2UpperCh(aSecond);
   aTime[7] = BCD2LowerCh(aSecond);

   if(aAP)
   {
      aTime[9] = 'P';
      aTime[10] = 'M';
   }
   else
   {
      aTime[9] = 'A';
      aTime[10] = 'M';
   }
```

```
Lcd_Set_Cursor(1,1);
Lcd_Write_String("Set Alarm       ");
Lcd_Set_Cursor(2,1);
Lcd_Write_String("Time: ");
Lcd_Write_String(aTime);

if(mt && t && kp)
{
   if(mt == 1 && t < '2')
   {
        aHour = BCD2Binary(aHr);
        temp = aHour % 10;

        aHour = (t - 48)*10 + temp;
        aHour = Binary2BCD(aHour);
        aHour = aHour | 0x40;
        writeDS3234(0x09, aHour);
   }
   else if(mt == 2)
   {
       aHour = BCD2Binary(aHr);
       aHour = aHour/10;

       aHour = aHour*10 + (t - 48);
       if(aHour < 13 && aHour > 0)
       {
          aHour = Binary2BCD(aHour);
          aHour = aHour | 0x40;
          writeDS3234(0x09, aHour);
       }
```

```
}
else if(mt == 3 && t < '6')
{
   aMinute = BCD2Binary(aMinute);
   temp = aMinute % 10;

   aMinute = (t - 48)*10 + temp;
   aMinute = Binary2BCD(aMinute);
   writeDS3234(0x08, aMinute);
}
else if(mt == 4)
{
   aMinute = BCD2Binary(aMinute);
   aMinute = aMinute/10;

   aMinute = aMinute*10 + (t - 48);
   if(aMinute < 60)
   {
      aMinute = Binary2BCD(aMinute);
      writeDS3234(0x08, aMinute);
   }
}
else if(mt == 5)
{
   aSecond = 0;
   writeDS3234(0x07, aSecond);
}
else if(mt == 6)
{
   aHour = readDS3234(0x09);
   if(aAP)
```

```
      {
        aHour = aHour & 0b11011111;
      }
      else
      {
        aHour = aHour | 0b00100000;
      }
      writeDS3234(0x09,aHour);
      mt = 0;
    }
    writeDS3234(0x0A,0x80);
    writeDS3234(0x0F,0xC8);
    writeDS3234(0x0E,0x1D);
  }

}
else if(m == 2)
{
  if(mt && t && kp)
  {
    if(mt == 1 && t < '2')
    {
       hour = BCD2Binary(hr);
       temp = hour % 10;

       hour = (t - 48)*10 + temp;
       hour = Binary2BCD(hour);
       hour = hour | 0x40;
       writeDS3234(2, hour);
    }
    else if(mt == 2)
```

```
{
   hour = BCD2Binary(hr);
   hour = hour/10;

   hour = hour*10 + (t - 48);
   if(hour < 13 && hour > 0)
   {
      hour = Binary2BCD(hour);
      hour = hour | 0x40;
      writeDS3234(2, hour);
   }
}
else if(mt == 3 && t < '6')
{
   minute = BCD2Binary(minute);
   temp = minute % 10;

   minute = (t - 48)*10 + temp;
   minute = Binary2BCD(minute);
   writeDS3234(1, minute);
}
else if(mt == 4)
{
   minute = BCD2Binary(minute);
   minute = minute/10;

   minute = minute*10 + (t - 48);
   if(minute < 60)
   {
      minute = Binary2BCD(minute);
      writeDS3234(1, minute);
```

```
        }
      }
      else if(mt == 5)
      {
        second = 0;
        writeDS3234(0, second);
      }
      else if(mt == 6)
      {
        hour = readDS3234(2);
        if(ap)
        {
          hour = hour & 0b11011111;
          writeDS3234(2,hour);
        }
        else
        {
          hour = hour | 0b00100000;
          writeDS3234(2,hour);
        }
        mt = 0;
      }
    }
    Lcd_Set_Cursor(1,1);
    Lcd_Write_String("Set Time          ");
    Lcd_Set_Cursor(2,1);
    Lcd_Write_String("Time: ");
    Lcd_Write_String(time);
  }
  else if(m == 3) //Date Settings
  {
```

```
if(mt && t && kp)
{
   if(mt == 1 && t < '4')
   {
      day = BCD2Binary(day);
      temp = day % 10;

      day = (t - 48)*10 + temp;

      day = Binary2BCD(day);
      writeDS3234(4, day);
   }
   else if(mt == 2)
   {
      day = BCD2Binary(day);
      day = day/10;

      day = day*10 + (t - 48);
      if(day < 32 && day > 0)
      {
         day = Binary2BCD(day);
         writeDS3234(4, day);
      }
   }
   else if(mt == 3 && t < '2')
   {
      month = BCD2Binary(month);
      temp = month % 10;

      month = (t - 48)*10 + temp;
   month = Binary2BCD(month);
   writeDS3234(5, month);
}
else if(mt == 4)
{
   month = BCD2Binary(month);
   month = month/10;

   month = month*10 + (t - 48);
   if(month < 13 && month > 0)
   {
      month = Binary2BCD(month);
      writeDS3234(5, month);
   }
}
else if(mt == 5)
{
   year = BCD2Binary(year);
   temp = year % 10;

   year = (t - 48)*10 + temp;
   year = Binary2BCD(year);

   writeDS3234(6, year);
}
else if(mt == 6)
{
   year = BCD2Binary(year);
   year = year/10;

   year = year*10 + (t - 48);
```

```
                year = Binary2BCD(year);


                writeDS3234(6, year);
              }
            }


            Lcd_Set_Cursor(1,1);
            Lcd_Write_String("Set Date          ");
            Lcd_Set_Cursor(2,1);
            Lcd_Write_String("Date: ");
            Lcd_Write_String(date);
          }


        }
      }
```