```
/*
 Example 13.1
 Create a digital LED display clock
 with alarm and snooze
 http://tronixstuff.com/tutorials > Chapter Thirteen
 based on code by Maurice Ribble
 17-4-2008 - http://www.glacialwanderer.com/hobbyrobotics

 */

#include "Wire.h"
#define DS1307_I2C_ADDRESS 0x68

int latchpin = 9; // connect to pin 12 on the '595
int clockpin = 7; // connect to pin 11 on the '595
int datapin = 10; // connect to pin 14 on the '595
int button1 = 8; // for the four menu buttons
int button2 = 2;
int button3 = 3;
int button4 = 4;
int alarmhour = 12;
int alarmminute = 0;
int alarmled=12;
int colonled=11;
int displayonoff=1; // display off = 0, on = 1
int alarmonoff=0; // alarm off = 0; on = 1
int lhd = 0;
int mhd = 0;
int mhd2 = 0;
int rhd = 0;
int exitmenu = 0;
int menuoption = 1;
int snooze = 0;
float a = 0;
int b = 0;
int c= 0;
float d = 0;
int leadingzero = 1; // 0 for no leading zereos, 1 for leading zeroes
int rnum = 0;
int zzz=0;
int segdisp[10] = {
  125,9,103,79,27,94,126,13,127,95}; // base 10 equivalents for digits 0~9
int posdisp[4] = {
  1,2,4,8}; // base 10 equivalents to close anodes on display 0~3 on module

void setup()
{
  byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
  pinMode(6, OUTPUT);
  pinMode(alarmled, OUTPUT);
  pinMode(latchpin, OUTPUT);
  pinMode(clockpin, OUTPUT);
  pinMode(colonled, OUTPUT);
  pinMode(datapin, OUTPUT);
  pinMode(button1, INPUT);
  pinMode(button2, INPUT);
  pinMode(button3, INPUT);
  pinMode(button4, INPUT);
  Wire.begin();
  Serial.begin(9600); // for debugging

  // Change these values to what you want to set your clock to.
  // You probably only want to set your clock once and then remove
  // the setDateDs1307 call.

  second = 0;
  minute = 42;
  hour = 23;
  dayOfWeek = 5;
  dayOfMonth = 20;
  month = 5;
  year = 10;
```

```
  // setDateDs1307(second, minute, hour, dayOfWeek, dayOfMonth, month, year);
}

// Convert normal decimal numbers to binary coded decimal
byte decToBcd(byte val)
{
  return ( (val/10*16) + (val%10) );
}

// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
  return ( (val/16*10) + (val%16) );
}

// 1) Sets the date and time on the ds1307
// 2) Starts the clock
// 3) Sets hour mode to 24 hour clock

// Assumes you're passing in valid numbers

void setDateDs1307(byte second,        // 0-59
byte minute,        // 0-59
byte hour,          // 1-23
byte dayOfWeek,     // 1-7
byte dayOfMonth,    // 1-28/29/30/31
byte month,         // 1-12
byte year)          // 0-99
{
  Wire.beginTransmission(DS1307_I2C_ADDRESS);
  Wire.send(0);
  Wire.send(decToBcd(second));    // 0 to bit 7 starts the clock
  Wire.send(decToBcd(minute));
  Wire.send(decToBcd(hour));
  Wire.send(decToBcd(dayOfWeek));
  Wire.send(decToBcd(dayOfMonth));
  Wire.send(decToBcd(month));
  Wire.send(decToBcd(year));
  Wire.send(0x10); // sends 0x10 (hex) 00010000 (binary) to control register - turns on square wave
  Wire.endTransmission();
}

void cleardisplay()
// turns off all segments of all digits
{
  for (int aa=0; aa<4; aa++)
  {
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, posdisp[aa]); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 0); // clears the digit
    digitalWrite(latchpin, HIGH);
  }
}

void onedigitnumber(int subject)
// displays a one-digit number on the display module with leading zeroes
{
  cleardisplay();
  if (leadingzero==1)
  {
    digitdisplay(0,0);
    digitdisplay(0,1);
    digitdisplay(0,2);
  }
  digitdisplay(subject,3);
}

void twodigitnumber(int subject)
// displays a two-digit number on the display module with leading zeroes
{
  cleardisplay();
  rhd = subject % 10;
```

```
  a = subject/10;
  lhd = int(a);
  if (leadingzero==1)
  {
    digitdisplay(0,0);
    digitdisplay(0,1);
  }
  digitdisplay(lhd,2);
  digitdisplay(rhd,3);
}

void threedigitnumber(int subject)
// displays a three-digit number on the display module with leading zeroes
{
  cleardisplay();
  a = subject/100;
  lhd = int(a);
  a = subject/10;
  b = int(a);
  mhd = b % 10;
  b=subject%100;
  rhd=b%10;
  if (leadingzero==1)
  {
    digitdisplay(0,0);
  }
  digitdisplay(lhd,1);
  digitdisplay(mhd,2);
  digitdisplay(rhd,3);
}

void displaynumber(int rawnumber, int cycles)
// takes an integer and displays it on our 4-digit LED display module
{
  for (int q=1; q<=cycles; q++)
  {
    if (rawnumber>=0 && rawnumber<10)
    {
      onedigitnumber(rawnumber);
    }
    else if (rawnumber>=10 && rawnumber<100)
    {
      twodigitnumber(rawnumber);
    }
    else if (rawnumber>=100 && rawnumber<1000)
    {
      threedigitnumber(rawnumber);
    }
    else if (rawnumber>=1000)
    {
      fourdigitnumber(rawnumber);
    }
  }
}

void fourdigitnumber(int subject)
// displays a four-digit number on the display module with leading zeros
{
  cleardisplay();
  a = subject/1000;
  lhd = int(a);
  b=lhd*1000;
  c=subject-b;
  a = c/100;
  mhd = int(a);
  a = c/10;
  b = int(a);
  mhd2 = b % 10;
  b=subject%1000;
  c=b%100;
  rhd=c%10;
  digitdisplay(lhd,0);
```

```
  digitdisplay(mhd,1);
  digitdisplay(mhd2,2);
  digitdisplay(rhd,3);
}

void digitdisplay(int digit, int location)
// displays "digit" on display "location" 0~3
{
  digitalWrite(latchpin, LOW);
  shiftOut(datapin, clockpin, MSBFIRST, posdisp[location]); // sets the digit to address
  shiftOut(datapin, clockpin, MSBFIRST, segdisp[digit]); // clears the digit
  digitalWrite(latchpin, HIGH);
}

// Gets the date and time from the ds1307
void getDateDs1307(byte *second,
byte *minute,
byte *hour,
byte *dayOfWeek,
byte *dayOfMonth,
byte *month,
byte *year)
{
  // Reset the register pointer
  Wire.beginTransmission(DS1307_I2C_ADDRESS);
  Wire.send(0);
  Wire.endTransmission();

  Wire.requestFrom(DS1307_I2C_ADDRESS, 7);

  // A few of these need masks because certain bits are control bits
  *second     = bcdToDec(Wire.receive() & 0x7f);
  *minute     = bcdToDec(Wire.receive());
  *hour       = bcdToDec(Wire.receive() & 0x3f);  // Need to change this if 12 hour am/pm
  *dayOfWeek  = bcdToDec(Wire.receive());
  *dayOfMonth = bcdToDec(Wire.receive());
  *month      = bcdToDec(Wire.receive());
  *year       = bcdToDec(Wire.receive());
}

void checkbuttons()
{
  if (digitalRead(button1)==HIGH)
    // otherwise, pressing button 1 opens the menu
  {
    delay(20);
    mainmenu();
    return;
  }
  if (digitalRead(button2)==HIGH) // has the user pressed button 2 to turn on the alarm?
  {
    if (displayonoff==1)
    {
      delay(5);
      alarmonoff=1;
      if (displayonoff==1)
      {
        digitalWrite(alarmled, HIGH);
      }
    }
  }
  if (digitalRead(button3)==HIGH)  // has the user pressed button 3 to turn off the alarm?
  {
    if (displayonoff==1) // don't let the user switch the alarm off when the display is off
    {

      delay(5);
      alarmonoff=0;
      digitalWrite(alarmled, LOW);
    }
  }
  if (digitalRead(button4)==HIGH) // has the user pressed button 4 to turn off display?
```

```
  {
    delay(5);
    switch(displayonoff)
    {
    case 1:
      displayonoff=0;
      cleardisplay();
      digitalWrite(colonled, LOW);
      digitalWrite(alarmled, LOW);
      delay(500); // for debounce
      break;
    case 0:
      displayonoff=1;
      delay(500); // for debounce
      break;
    }
  }
}

void checkalarm()
{
  byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
  getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);
  if (alarmonoff==1)
    // if the alarm is on, check the alarm time against the real time, and sound alarm, etc if alarm
time
  {
    if (displayonoff==1)
    {
      digitalWrite(alarmled, HIGH);
    }
  }
  if (alarmonoff==0)
    // if the alarm is off, make sure the indicator LED is off
  {
    digitalWrite(alarmled, LOW);
  }
  if (alarmhour==hour && alarmminute==minute)
    // wake up!
  {
    analogWrite(6, 128); // connect a buzzer, relay, LED, whatever you want to activate when the alarm
goes off
    digitalWrite(colonled,LOW);
    cleardisplay();

    while (digitalRead(button4)==LOW) // user presses button 4 to turn off buzzer
    {
      delay(1);
    }
    digitalWrite(6, LOW); // turns off buzzer
    exitmenu=0;
    snooze=0;
    delay(1000); // debounce

    do
    {
      // display "Sno?" for Snooze?
      digitalWrite(latchpin, LOW);
      shiftOut(datapin, clockpin, MSBFIRST, 1);
      shiftOut(datapin, clockpin, MSBFIRST, 94);
      digitalWrite(latchpin, HIGH);
      digitalWrite(latchpin, LOW);
      shiftOut(datapin, clockpin, MSBFIRST, 2);
      shiftOut(datapin, clockpin, MSBFIRST, 42);
      digitalWrite(latchpin, HIGH);
      digitalWrite(latchpin, LOW);
      shiftOut(datapin, clockpin, MSBFIRST, 4);
      shiftOut(datapin, clockpin, MSBFIRST, 106);
      digitalWrite(latchpin, HIGH);
      digitalWrite(latchpin, LOW);
      shiftOut(datapin, clockpin, MSBFIRST, 8);
      shiftOut(datapin, clockpin, MSBFIRST, 39);
```

```
      digitalWrite(latchpin, HIGH);

      if (digitalRead(button1)==HIGH) // turn on snooze
      {
        snooze=1;
        alarmonoff=1;
        exitmenu=1;
      }
      if (digitalRead(button4)==HIGH) // no snooze, just turn off alarm
      {
        exitmenu=1;
        alarmonoff=0;
      }
    }
    while (exitmenu==0);

    if (snooze==1)
    // recalculate alarm time and turn alarm on
    {
      alarmonoff=1; // ensure alarm is still on
      alarmminute=alarmminute+10; // add ten minutes to alarm time
      if (alarmminute>59)
      {
        alarmminute=alarmminute-60;
        alarmhour=alarmhour+1;
        if (alarmhour>23)
        {
          alarmhour=alarmhour-24;
        }
      }
    }

    if (alarmonoff==0)
    {
      digitalWrite(colonled,LOW);
      // blinks "OFF" on display for about fifty seconds
      for (int ii=0; ii<50; ii++)
      {
        for (int i=0; i<1000; i++)
        {
          digitalWrite(latchpin, LOW);
          shiftOut(datapin, clockpin, MSBFIRST, 1); // sets the digit to address
          shiftOut(datapin, clockpin, MSBFIRST, 125); // clears the digit
          digitalWrite(latchpin, HIGH);
          digitalWrite(latchpin, LOW);
          shiftOut(datapin, clockpin, MSBFIRST, 2); // sets the digit to address
          shiftOut(datapin, clockpin, MSBFIRST, 54); // clears the digit
          digitalWrite(latchpin, HIGH);
          digitalWrite(latchpin, LOW);
          shiftOut(datapin, clockpin, MSBFIRST, 4); // sets the digit to address
          shiftOut(datapin, clockpin, MSBFIRST, 54); // clears the digit
          digitalWrite(latchpin, HIGH);
          digitalWrite(latchpin, LOW);
          shiftOut(datapin, clockpin, MSBFIRST, 8); // sets the digit to address
          shiftOut(datapin, clockpin, MSBFIRST, 0); // clears the digit
          digitalWrite(latchpin, HIGH);
        }
        delay(100);
        cleardisplay();
        delay(100);
      }
    }
  }
}

void showtime() // just shows the time on the display
{
  byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
  getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);
  if (bcdToDec(hour)<1)
  {
    zzz=minute;
```

```
  }
  else if (bcdToDec(hour)>=1)
  {
    zzz=hour*100;
    zzz=zzz+minute;
  }
  displaynumber(zzz,5);// only display the time for 5 milliseconds, gives the void loop adequate
button response time
  digitalWrite(colonled, HIGH);
}


void mainmenu()
{
  menuoption=1;
  delay(100);
  digitalWrite(colonled, LOW);
  digitalWrite(alarmled, LOW);
  while (exitmenu==0)
  {
    if (digitalRead(button1)==HIGH) // press button 1 to exit menu
    {
      delay(100);
      exitmenu=1;
    }
    if (digitalRead(button2)==HIGH) // press button 2 to set time
    {
      delay(100);
      menuoption=1;
    }
    if (digitalRead(button3)==HIGH) // press button 3 to set alarm
    {
      delay(100);
      menuoption=2;
    }
    if (digitalRead(button4)==HIGH) // press button 4 to select option
    {
      delay(100);
      executeoption(menuoption);
    }
    displaymenuoption(menuoption);
  }
  exitmenu=0;
  delay(100);
}

void executeoption(int z)
{
  if (z==1)
  {
    settime();
  }
  if (z==2)
  {
    setalarm();
  }
}

void settime()
{
  byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
  second = 0;// dummy data to store in DS1307
  dayOfWeek = 1;
  dayOfMonth = 1;
  hour=12;
  minute=0;
  month = 1;
  year = 1;
  cleardisplay();
  digitalWrite(colonled, HIGH);
  digitalWrite(alarmled, LOW);
  while (digitalRead(button4)==LOW)
```

```
  {
    zzz=(hour*100);
    zzz=zzz+minute;
    displaynumber(zzz,5);// only display the time for 5 milliseconds, gives the void loop adequate
button response time

    if (digitalRead(button1)==HIGH)
    {
      delay(100);
      hour++;
      if (hour>23)
      {
        hour=0;
      }
    }

    if (digitalRead(button2)==HIGH)
    {
      delay(100);
      minute=minute+10;
      if (minute>50)
      {
        minute=50;
      }
    }

    if (digitalRead(3)==HIGH)
    {
      delay(100);
      minute++;
      if (minute>59)
      {
        minute=0;
      }
    }
  }
  setDateDs1307(second, minute, hour, dayOfWeek, dayOfMonth, month, year);
  delay(100);
}

void setalarm()
{
  cleardisplay();
  digitalWrite(colonled, HIGH);
  digitalWrite(alarmled, LOW);
  while (digitalRead(button4)==LOW)
  {
    zzz=(alarmhour*100);
    zzz=zzz+alarmminute;
    displaynumber(zzz,5);// only display the time for 5 milliseconds, gives the void loop adequate
button response time
    if (digitalRead(button1)==HIGH)
    {
      delay(100);
      alarmhour++;
      if (alarmhour>23)
      {
        alarmhour=0;
      }
    }
    if (digitalRead(button2)==HIGH)
    {
      delay(100);
      alarmminute=alarmminute+10;
      if (alarmminute>50)
      {
        alarmminute=50;
      }
    }
    if (digitalRead(3)==HIGH)
    {
      delay(100);
```

```
      alarmminute++;
      if (alarmminute>59)
      {
        alarmminute=0;
      }
    }
  }
  delay(100);
}


void displaymenuoption(int z)
{
  if (z==1)
  {
    // display "Ti ?"
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 1); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 13); // clears the digit
    digitalWrite(latchpin, HIGH);
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 2); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 32); // clears the digit
    digitalWrite(latchpin, HIGH);
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 4); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 0); // clears the digit
    digitalWrite(latchpin, HIGH);
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 8); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 39); // clears the digit
    digitalWrite(latchpin, HIGH);
  }
  if (z==2)
  {
    // display "Al ?"
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 1); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 63); // clears the digit
    digitalWrite(latchpin, HIGH);
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 2); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 96); // clears the digit
    digitalWrite(latchpin, HIGH);
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 4); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 0); // clears the digit
    digitalWrite(latchpin, HIGH);
    digitalWrite(latchpin, LOW);
    shiftOut(datapin, clockpin, MSBFIRST, 8); // sets the digit to address
    shiftOut(datapin, clockpin, MSBFIRST, 39); // clears the digit
    digitalWrite(latchpin, HIGH);
  }
}

void loop()
{
  if (displayonoff==1)
  {
    showtime();
  }

  checkbuttons(); // function to control menus etc.
  checkalarm(); // is it alarm time?
}
```