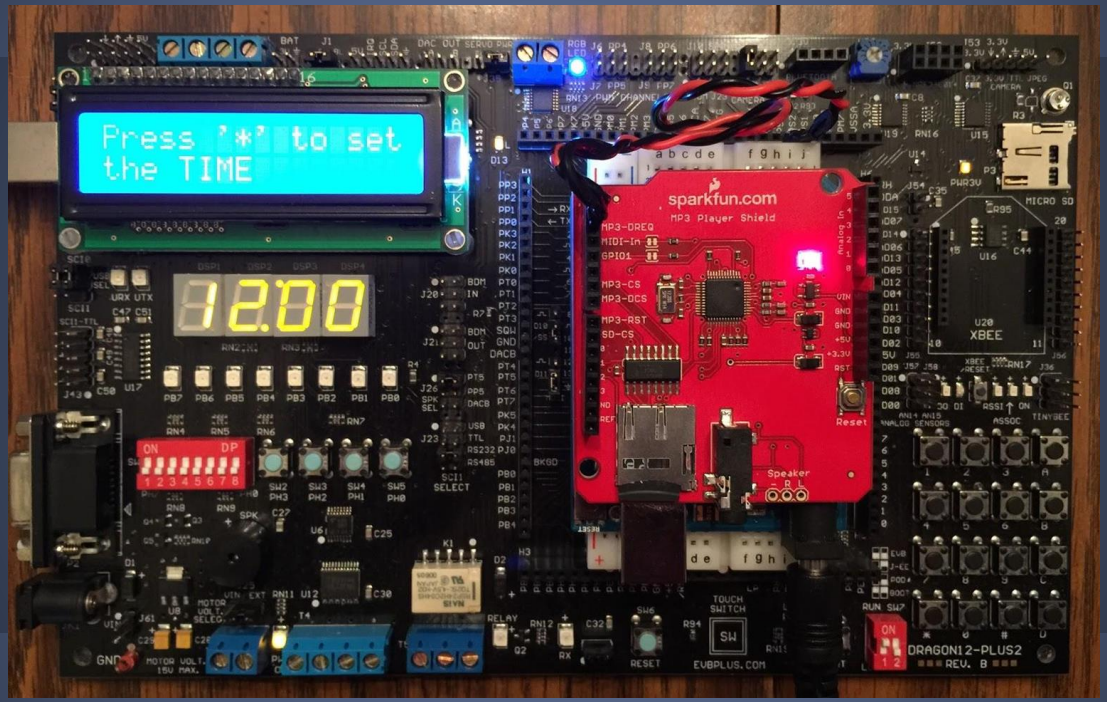
# ECE 470 Final Project




Digital Alarm Clock

# Project Members:

**Binh Ton**

**Matt Screws**

**Adam Schrader**

# Contents

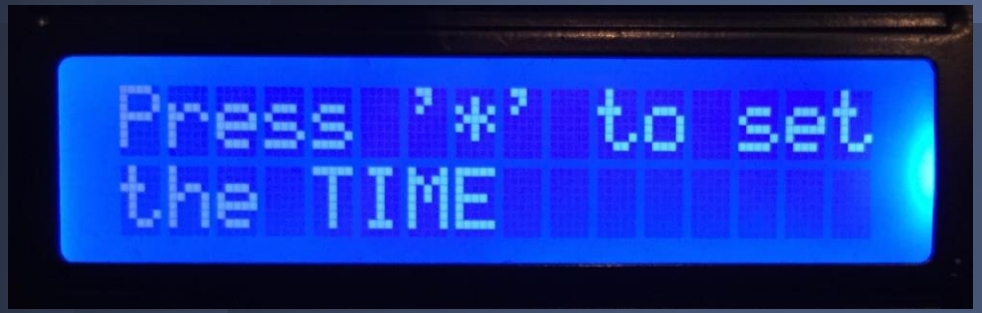
- Overview of Operation
  - LCD Menus
  - Hex Keypad Input
  - 7 Segment and RGB LED
  - Arduino Interface

# Contents

- Coding
  - LCD Menus
  - Hex Keypad
  - Alarm Trigger
  - Arduino
  - Clock Timer
  - 7 Segment Display and RGB LED Operation

# Overview of Operation

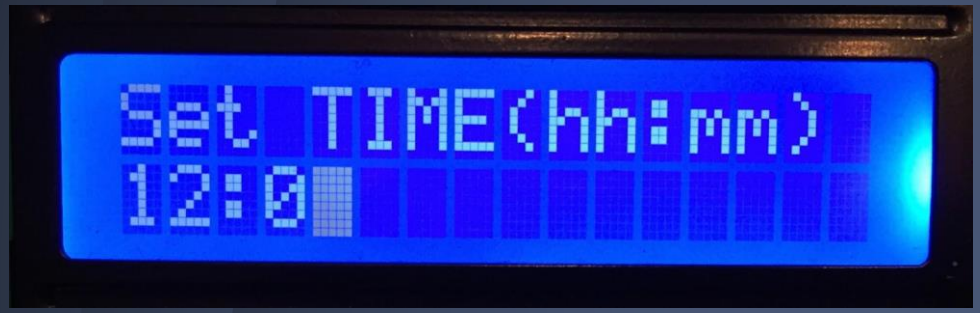
# LCD Menus



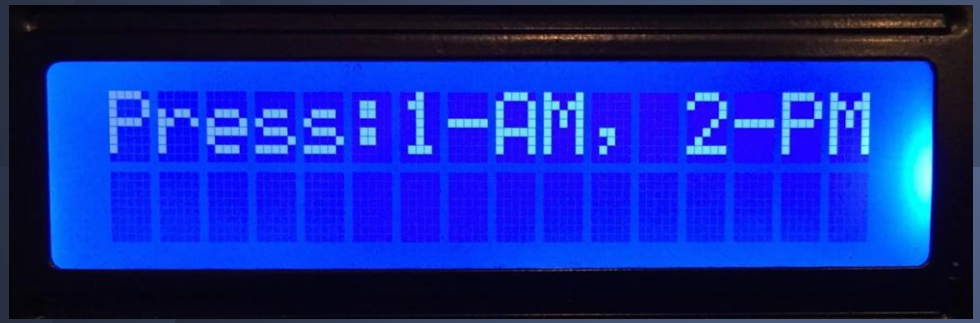

Set Time


Set Alarm

# LCD Menus Cont.

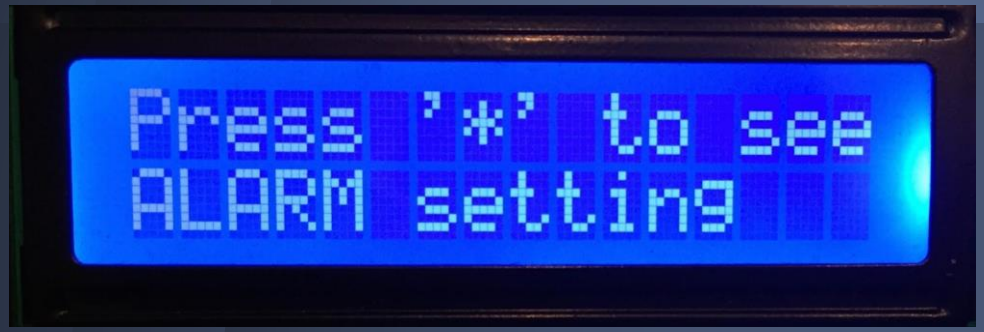



Setting the time and alarm




AM/PM Selection

# LCD Menus Cont.




View Alarm Status


Cancel Alarm

# LCD Menus Cont.

## Help Menu




# HEX Keypad Input


TIME/ALARM ENTRY PAD

UP

DOWN

BACKSPACE

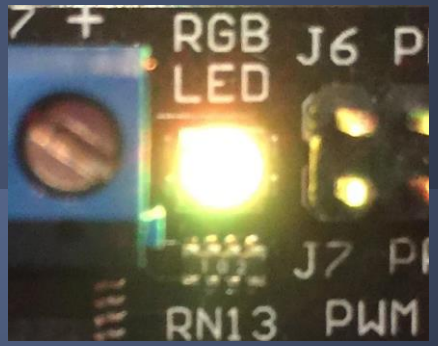
ENTER

CANCEL

# 7-seg and RGB LED


Clock Display


AM
Indicator



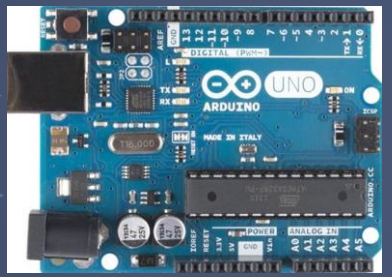

PM
Indicator

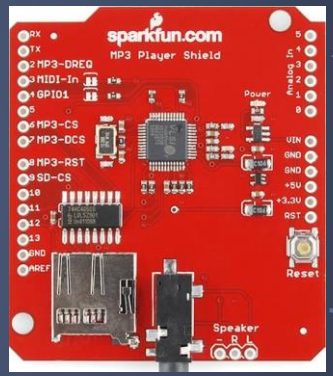
# Arduino Interface

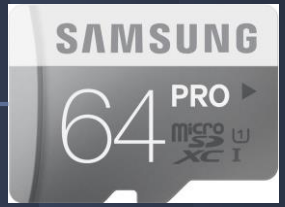
HCS12

ALARM! Play random_mp3()

STOP! Send char "s"




# Coding

# LCD Coding

```
c = keyscan();

  if (c == UP & enter_selected == 0)
  {
    switch(index)
    {
      case 0:
        index++;
        LCD_display(index);
        break;
      case 1:
        index++;
        LCD_display(index);
        break;
      case 2:
        index++;
        LCD_display(index);
        break;
      case 3:
        index++;
        LCD_display(index);
        break;
      case 4: break;
    }
  }
```

```
void LCD_display(int index)
{
  switch(index)
  {
    case 0:
      LCD_time_menu();
      break;
    case 1:
      LCD_alarm_menu();
      break;
    case 2:
      LCD_alarm_status_menu();
      break;
    case 3:
      LCD_alarm_cancel_menu();
      break;
    case 4:
      LCD_help_menu();
      break;
  }
}
```

```
if (c == ENTER) // if user pressed E for ENTER
{
  switch(index) // case to display user input menu based on index
  {
    case 0: // LCD_time_menu
      time_display(); // display time input menu
      S_CLOCK = 1; // set clock flag = 1 since we are setting the time
      enter_selected = 1; // enter pressed --> flag = 1
      get_time(); // function called to get time
      LCD_display(index); // display LCD_time_menu
      enter_selected = 0; // enter exited --> flag = 0
      break;
    case 1: // LCD_alarm_menu
      alarm_display(); // display alarm input menu
      S_CLOCK = 0; // set clock flag = 0 since we are setting the alarm
      enter_selected = 1; // enter pressed --> flag = 1
      get_time(); // function called to get alarm
      LCD_display(index); // display LCD_alarm_menu
      enter_selected = 0; // enter exited --> flag = 0
      break;
```

# Clock Timer

```
interrupt void oc4ISR(void){
  TC4 = TC4 + MCYCLES;
  ms250_cnt += 1;
  if(ms250_cnt == 4){
    mcount += 1;
    ms250_cnt = 0;
    if(mcount == 60){
      min++;
      mcount = 0;
    }
  }
}
```

# 7 Segment Display/RGB LED

```
interrupt void oc6ISR(void){
  TC6 = TC6 + del;
  switch(disp){

    case 0:
       seg7dec(min,3);
       disp = 1;
       break;
    case 1:
       seg7d(tmin,2);
       disp = 2;
       break;
    case 2:
       seg7d(hr,1);
       disp = 3;
       break;
    case 3:
       if(thr > 0){
         seg7dec(thr,0);
         disp = 0;
         break;
       }
       else{
         disp = 0;
         break;
       }
  }
}
```

Displaying the Segments

```
void interrupt 7 handler(){


  if(min == 10){
    tmin++;
    min = 0;
  }
  if(tmin == 6){
    hr++;
    tmin = 0;
  }
  if(hr == 3){
    if(thr == 1){
      thr = 0;
      hr = 1;
    }
  }
  if((thr == 1) && (hr == 1)){
    AP_flag = 1;
  }
  if((thr == 1) && (hr == 2) && (AP_flag == 1)){
    AmPm = !AmPm;
    AP_flag = !AP_flag;
  }
  if(hr == 10){
    if(thr == 1){
      thr = 0;
      hr = 1;
    }
    else{
      thr++;
      hr = 0;
    }
  }
  if(AmPm == 0){
    PTP &= 0x0F;
    PTP |= 0x50;
  }
  else{
    PTP &= 0x0F;
    PTP |= 0x20;
  }

clear_RTI_flag();
alarm_trig();

}
```

Logic Behind the Segments

# HEX Keypad Coding

```
c = keyscan();

  if (c == UP & enter_selected == 0)
  {
    switch(index)
    {
      case 0:
        index++;
        LCD_display(index);
        break;
      case 1:
```

```
case 4:
        c = getkey();
        wait_keyup();
        while(c <=11){
          c = getkey();
          wait_keyup();
        }
        if(c == BACK){
            set_lcd_addr(0x44);
            data8(SPACE);
            set_lcd_addr(0x44);
            time_index--;//set time_index to 1
            instr8(0x0D);
            break;
          }
          if(c == CANCEL){
            instr8(0x0C);
            time_index = 8;
            break;

          }
          else{
            if(S_CLOCK == 1){
              min  = smin;
              tmin = stmin;
              hr   = shr;
              thr  = sthr;
            }
            else{
              amin = smin;
              atmin = stmin;
              ahr   = shr;
              athr  = sthr;
            }
            time_index = 5;
            break;
          }
```

# Alarm Coding

```
void alarm_trig(void){
   char alarm_sound;

  if((thr == athr) && (hr == ahr) && (tmin == atmin) && (min == amin) && (AmPm == A_AmPm)){
    alarm_sound = 1;
  }
  else{
    alarm_sound = 0;
    alarm_triggered_flag2 = 0;
  }
  if((alarm_set_flag == 1) && (alarm_triggered_flag2 == 1)){
    if(alarm_sound == 1 ){
        alarm_trig_flag = 0;
        alarm_triggered_flag2 = 1;
      }
      else{
        alarm_trig_flag = 0;
      }
  }
  if((alarm_set_flag == 1) && (alarm_triggered_flag2 == 0)){
    if(alarm_sound == 1 ){
      alarm_trig_flag = 1;
      alarm_triggered_flag2 = 1;
    }
    else{
      alarm_trig_flag = 0;
    }
  }
  if(alarm_set_flag == 0){
        alarm_trig_flag = 0;
        alarm_triggered_flag2 = 0;
  }
}
```

# Arduino Coding

```
track = random_mp3(); // call random_mp3 function to get random track numer to play
outchar1(track); // send track number over SCI1 to Arduino Uno
alarm_trig_display(); // alarm has triggered so display appropriate menu
alarm_trig_flag = 0;
c = getkey();
wait_keyup();
while(!(c == CANCEL)){ // waits for user to cancel the alarm before moving on
  c = getkey();
  wait_keyup();
}
outchar1(STOP); // user has cancelled alarm so send STOP char to stop alarm
LCD_display(index); // display previous menu
```

```
char random_mp3(void)
{
    int result = 0;
    char track_number;

    srand(TC6+TC4+TCNT);
    result = (rand()%8)+1;
    track_number = result + 0x30;
    return track_number;
}
```

# References


1. Budakoti, A. (2014, May 9). How to generate a random number in C? Retrieved from http://stackoverflow.com/questions/822323/how-to-generate-a-random-number-in-c

2. Haskell, R., & Hanna, D. (2008). Hex Keypad. In *Learning by example using C: Programming DRAGON12-Plus using CodeWarrior* (Second ed., p. 28). Rochester: LBE Book

3. Huang, H. (2006). *The HCS12/9S12: An introduction to software and hardware interfacing*(Second ed.). Clifton Park, NY: Delmar/Thomson Learning.

4. Porter, B. (2012, January 28). Sparkfun MP3 Shield Arduino Library. Retrieved December 2, 2014, from http://www.billporter.info/2012/01/28/sparkfun-mp3-shield-arduino-library/