EEL 4924 Electrical Engineering Design
(Senior Design)

Final Report

21 April 2009

Project Title: Protean Alarm Clock

Team Name: Early Bird

Team Members:
Name: Robin Whittle
Email: rwhittle@ufl.edu
Phone: (410) 610-8066

Team: Early Bird

Project Abstract:
The Protean Alarm Clock is a stand-alone alarm clock with dynamic features designed to wake the user quickly and effectively. The alarm's sound is unpredictable, and the user is forced to think and interact with the clock to turn the alarm off. The technical challenges of the project were primarily software-related: writing the programs to allow the user to set the clock and the alarm, as well as programs to control the alarm and the routines to allow the user to shut it off. Hardware challenges will dealt with the implementation of the speaker and display screen for the clock and a variety of components used in the disabling routines.

Team: Early Bird

Table of Contents

Tables and Figures

Features:

This alarm clock is intended for people who are able to sleep through a typical alarm or disable one in their sleep when the process becomes too routine. Most alarm clocks go off each time with the same sound, the notable exceptions being clocks that use a radio, CD, or mp3 player as the alarm. This clock combines the obnoxiousness of a standard alarm with the spontaneity of a radio alarm by randomly selecting from a set of programmed patterns of tones stored within the clock.

The clock is also intended to ensure that the user actually wakes up, rather than simply shutting off the alarm and falling asleep once again. Various methods have been used in other products, including clocks that roll or fly away and must be chased, but this clock requires cognitive processes to disable the alarm. The user is required to identify randomly selected color, then answer three random multiple choice questions on various subjects. The disabling sequence will change each time the alarm goes off, and an incorrect answer will lock the user out of the disabling function for 10 seconds before resetting the sequence and beginning again from a new color-identification question.

Similar Products:

Various manufacturers have alarm clocks that use CD or mp3 players as the alarm, including the Sony and iLive models shown below. These products have an element of randomization, but rely on "alarms" – that is, music – stored externally, either on CDs or mp3 players, and are therefore less obnoxious than standard alarm clocks.

Figure 1. Sony ICF-CD815 AM/FM Stereo CD Clock Radio with Dual Alarm
Source: http://www.amazon.com/Sony-ICF-CD815-Stereo-Clock-Radio/dp/B000MXYPYW/ref=pd_bbs_1?ie=UTF8&s=electronics&qid=1233046687&sr=8-1

Figure 2. iLive IC618B Clock Radio
Source: http://www.amazon.com/iLive-IC618B-Enhanced-Docking-Station/dp/B0015AOHDK/ref=sr_1_11?ie=UTF8&s=electronics&qid=1233047025&sr=1-11

Both the Clocky and the Flying Alarm Clock (pictured below) require the user to chase them in order to disable the alarm. While this is undoubtedly more complicated than simply pressing a button, it does not require fine motor skills or cognitive processes, and hard-core sleepers can still shut them off without waking up.

Figure 3. Clocky

Source: http://www.uberreview.com/2007/04/look-at-clocky-go.htm

Figure 4. Flying Alarm Clock

Source: http://www.slashgear.com/flying-alarm-clock-gets-you-up-142479/

Concept Selection:

The central components of the clock are the visual and aural outputs. Also necessary were methods of collecting user inputs and a processor to run the program.

An Atmega32 was chosen for the processor for its diverse functionality, relatively large number of I/O pins – which would be needed for the various interactive features, sufficient amount of memory, and satisfactory processing speed – which was important for a project that features both constant real time interrupts and spontaneous user input.

For the visual display, several options were available. A typical clock has only a four-character numeric display, but the disarming function requires a method of displaying text, so an LCD with at least two lines of characters was necessary. A large GLCD would not be appropriate, and, as long as the disarming questions were written to be short, a simple 20x2 display would be sufficient for the interactive portion of the alarm clock. The characters on such a display are relatively small, however, and a 7-segment LED display would be more suited to a clock, so such a display was originally included in the design. Four 1.5 cm (5/8 in) tall 7-seg LEDs were to be controlled by a CPLD. Unfortunately, the logic design required a CPLD with at least 58 I/O pins, and it was decided that the time necessary to design and populate a PCB to run such a display was better spent on other aspects of the project. The second, redundant display was abandoned, and the clock display is entirely contained on a 20x2 LCD.

Options for the aural output were also varied. Integration with a radio or mp3 player was ruled out early in the design process because the alarm clock was intended to be completely self-contained and music is not jarring enough for the purposes of this alarm. In order to maximize irritation, a piezo-electric buzzer rated at up to 86 dB was chosen. An external EEPROM was considered to store short songs, which would be sent to the speaker by way of a latch and D/A converter combination, but in the end, the PWM output function of the processor's Timer/Counter0 Interrupt register was used in order to optimize the number of output pins needed on the processor. Different alarms were created by changing the pre-scale factor of the interrupt each second, and the output of the interrupt is sent to the buzzer after being amplified by two inverting op amps.

Several different input methods were considered for the disarming sequence. Simple push buttons were chosen for the multiple-choice portion and for setting the clock and alarm. Toggle switches, slide switches, and levers were considered for their different demands on fine motor skills, and it was eventually decided to use a potentiometer connected to a voltage comparator and ask the user to turn the dial to a specific level in order to proceed to the next step in the disarming process. The dial was designed and implemented on the breadboard, but unfortunately a set of traces were ripped off of the PCB at the eleventh hour of project finalization, and the dial could not be included in the final design. Instead, another set of multiple choice questions was added to the program.

Two high-level system diagrams are below, showing the original design with separate displays, EEPROM, and dial (Figure 5) and the final design (Figure 6):

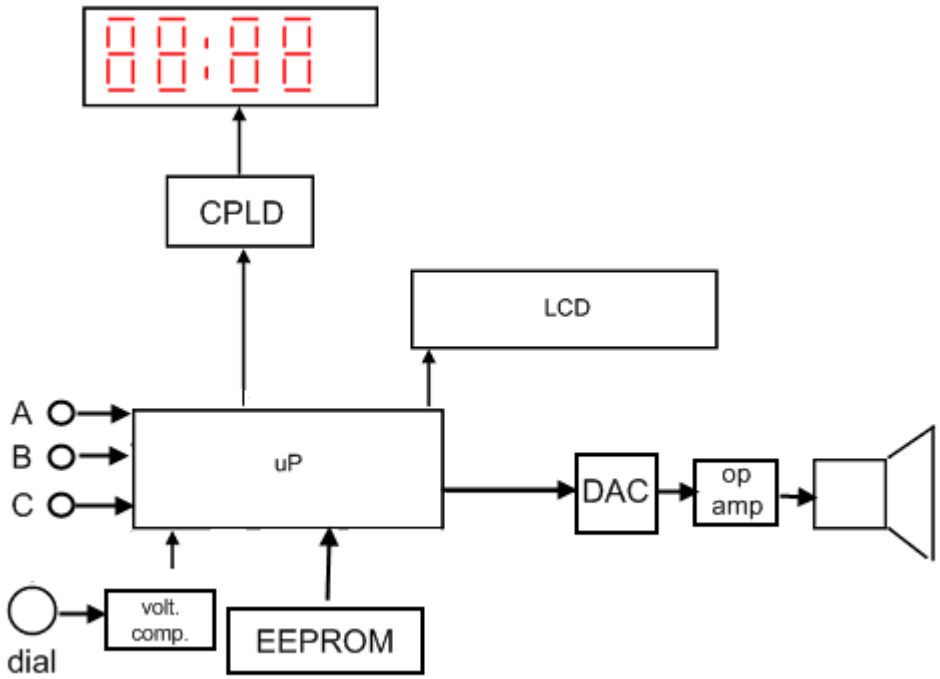

Figure 5. Original design with two displays

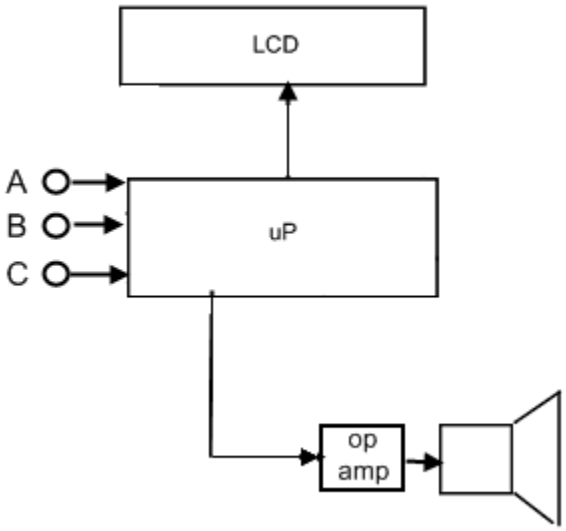

Figure 6. Final design

Project Architecture:

Figure 7 shows the program design:

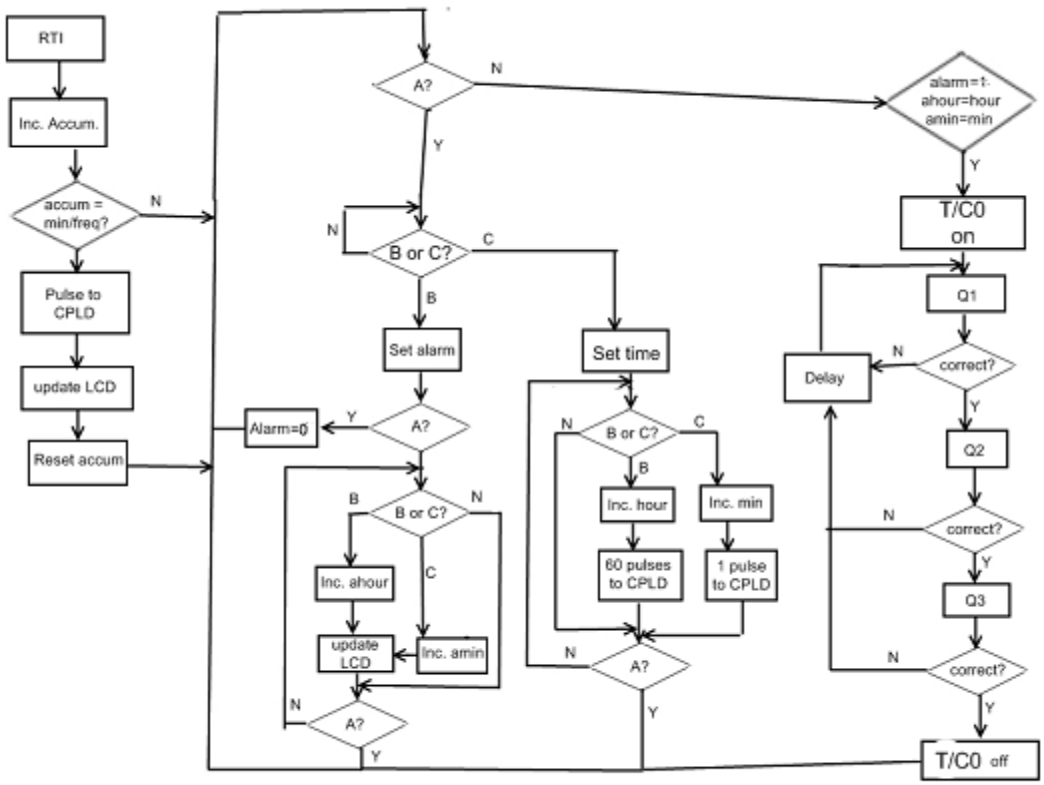

Figure 7. Software flowchart

When turned on, the LCD displays the time as 00:00 (midnight on a 24-hour clock) and prompts the user to press the "A" button to set the time or alarm. By following on-screen instructions, the user sets these values, which are updated appropriately on the display. The Timer/Counter1 interrupt is used as a real-time clock with interrupts firing every second. When 60 interrupts have occurred, the LCD is updated with the new time. In the original design with the 7-segment display, a pulse was sent to the CPLD each minute, which acted as the clock and triggered the logic circuits to increase the time appropriately.

When the clock time is equal to the set alarm time, a random number – determined from bits off of the counter and second count at the time – is chosen, which determines the pattern of frequencies sent to the buzzer. The PWM interrupt is enabled, and the pre-scaler is changed according to the alarm selected. A tri-color LED is turned on – the color having been selected randomly – and the user is asked to identify the color, as shown in Figure 8. If the answer is correct, an addition or multiplication question is asked, followed by two trivia questions. If at any point an incorrect answer is given, the user must wait ten seconds, then begin the disarming process again with new questions. Only when all four questions are answered correctly is the PWM interrupt cleared and the speaker is shut off; however, the alarm remains on for at least one minute, so, in the event that it is disabled before a full minute has passed, a new alarm will sound and the disarming sequence must be repeated.

Team: Early Bird

Figure 8. Color identification

Bill of Materials:

The cost of the prototype is itemized in Table 1 below:

Atmega32	$ 0.00
LCD	$ 5.00
buzzer	$ 1.99
batteries	$ 5.00
LM324	$ 0.50
LM358	$ 0.40
buttons	$ 1.50
switch	$ 0.95
tri-color LED	$ 2.99
potentiometers	$ 2.45
sockets	$ 3.94
resistors, etc	$ 1.50
Total	$26.22

Table 1. Final Cost

Team: Early Bird

Timeline:

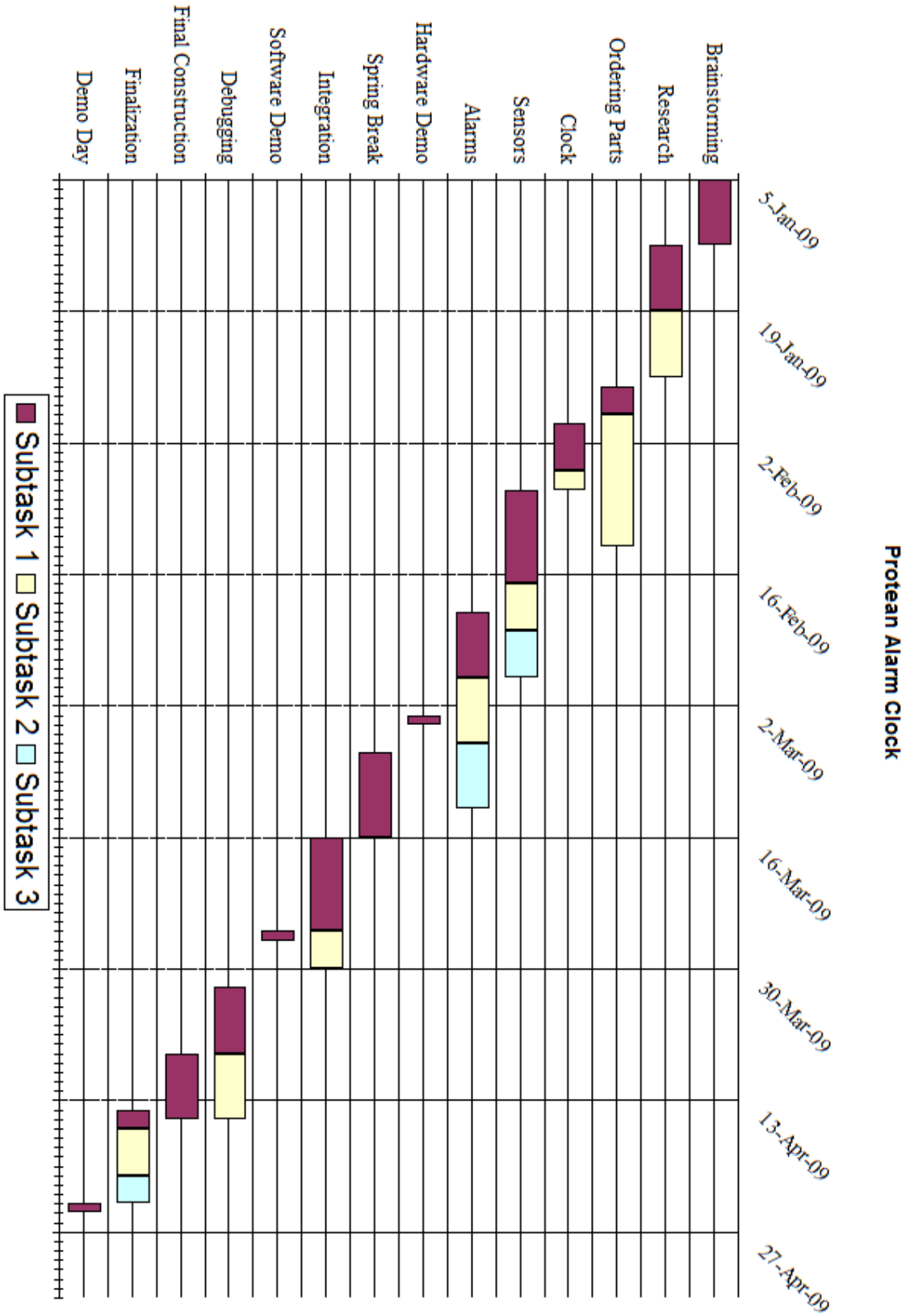

Table 2. Gantt Chart

Appendix – Code:

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include "LCD.h"

/*      Atmega32        */
/*                      */
/*                      */
/* A0   Out     Buzzer */
/* A1   Out     CPLD I */
/* A2   Out  CPLD Reset */
/* A3                   */
/* A4                   */
/* A5   In    button C */
/* A6   In    button B */
/* A7   In    button A */
/*                      */
/* B0   Out       LED1 */
/* B1   Out       LED2 */
/* B2   Out       LED3 */
/* B3                   */
/* B4            MOSI */
/* B5            MISO */
/* B6             SCK */
/* B7           /Reset */
/*                      */
/* C0   In       Dial4 */
/* C1   In       Dial3 */
/* C2            none */
/* C3            none */
/* C4            none */
/* C5            none */
/* C6   In       Dial2 */
/* C7   In       Dial1 */
/*                      */
/* D0   Out       LCD7 */
/* D1   Out       LCD6 */
/* D2   Out       LCD5 */
/* D3   Out       LCD4 */
/* D4   Out            */
/* D5   Out            */
/* D6   Out     LCD RS */
/* D7   Out      LCD E */
/*                      */

int min;  // Will use in main, RTI ISR, and Update LCD
int hour;
```

```
 int amin;
 int ahour;
 int aset;
 int sec;  // Will use in RTI ISR and UpdateLCD
 int alarmactive;
 int alarmsel;

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

int random(int min, int max)  // Generates a "random" number between 0 and 7 by reading the counter
{
 int randint = 0;
 do
    {
    randint = (TCNT1 & 0x0E)>>1;   // Gets three bits of counter
    _delay_us(TCNT1 & 0x07);
            } while ((randint < min) || (randint > max));  // Checks to make sure int is within specified
bounds
 return(randint);
}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

int updateLCD(int min, int hour, int amin, int ahour, int aset)
{
 LCD_ADDR(0x00);
 LCD_NUMBER(hour/10);
 LCD_NUMBER(hour%10);
 if (sec%2 == 0)            // pulse ":" on the second
    {
    LCD_CHARACTER(':');
            }
 else
    {
    LCD_CHARACTER(' ');
            }
 LCD_NUMBER(min/10);
 LCD_NUMBER(min%10);
 if (aset == 1)
    {
    LCD_STRING("   Alarm: ");
            LCD_NUMBER(ahour/10);
            LCD_NUMBER(ahour%10);
            LCD_CHARACTER(':');
            LCD_NUMBER(amin/10);
            LCD_NUMBER(amin%10);
            }
 else
    {
```

```
        LCD_STRING("              ");
        }
 return(0);
}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

ISR(TIMER1_COMPA_vect)
{
 sec+=1;
 if (sec == 60)
   {
   sec = 0;
//        PORTA = PORTA & 0x02;    // Pulse to CPLD
   if (min < 59)
     {
          min += 1;
     }
   else
          {
          min = 0;
          if (hour < 23)
            {
                 hour += 1;
                 }
     else
            {
                 hour = 0;
                 }
          }
        }
 if (alarmactive == 0)
   {
   updateLCD(min, hour, amin, ahour, aset);
   LCD_ADDR(0x40);
        }
 else
   {
   TIMSK = 0x90;
   TCCR0 = 0x6A;
   if (alarmsel == 0)
     {
          if ((sec%2) == 0)
            {
                 TCCR0 = 0x6A;
                 }
     else
            {
                 TCCR0 = 0x00;
```

```
                }
        }
else if (alarmsel == 1)
  {
        if ((sec%2) == 0)
          {
                TCCR0 = 0x6A;
                }
  else
          {
                TCCR0 = 0x6B;
                }
        }
else if (alarmsel == 2)
        {
        if ((sec%4) == 0)
          {
                TCCR0 = 0x6A;
                }
  else if ((sec%4) == 2)
          {
                TCCR0 = 0x6C;
                }
  else
          {
                TCCR0 = 0x6B;
                }
  }
else if (alarmsel == 3)
  {
        if (((sec%6)%2) == 1)
          {
                TCCR0 = 0x6A;
                }
  else if ((sec%6) == 4)
          {
                TCCR0 = 0x69;
                }
  else
          {
                TCCR0 = 0x6B;
                }
        }
else if (alarmsel == 4)
  {
        if (((sec%8) == 1)||((sec%8) == 7))
          {
                TCCR0 = 0x69;
                }
```

```
      else if ((sec%8) == 0)
            {
                  TCCR0 = 0x00;
                  }
      else if (((sec%8) == 2) || ((sec%8) == 6))
            {
                  TCCR0 = 0x6A;
                  }
      else if ((sec%8) == 4)
            {
                  TCCR0 = 0x6C;
                  }
      else
            {
                  TCCR0 = 0x6B;
                  }
            }
   else
      {
            if ((sec%4) == 2)
            {
                  TCCR0 = 0x6C;
                  }
      else if ((sec%4) == 3)
            {
                  TCCR0 = 0x6B;
                  }
      else
            {
                  TCCR0 = 0x6A;
                  }
            }
   }
}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

int question1()
{
 PORTA = 0xF0;
 int colorsel = random(0,3);    // selects LED color
 LCD_COMMAND(0x01);
 LCD_STRING("     The LED is:    ");
 LCD_ADDR(0x40);                    // Send cursor to address 0x40 (second row)
 _delay_ms(500);                    // Wait for 500 ms
 if (colorsel < 2)
    {
    LCD_STRING("A)Red B)Blue C)Green");
          if (colorsel == 0)
```

```
    {
        PORTA = 0xF0;
    PORTB = 0xFE;  // Turn on red
        do
          {
              if ((PINA & 0x80) == 0x00)  // Button A
                {
                    PORTB = 0xFF;
                    return(1);
                    }
      else if ((PINA & 0x40) == (0x00))  // Button B
                {
                    PORTB = 0xFF;
                    return(0);
                    }
      else if ((PINA & 0x20) == (0x00))  // Button C
                {
                    PORTB = 0xFF;
                    return(0);
                    }
      } while (1==1);
        }
  if (colorsel == 1)
    {
        PORTA = 0xF0;
        PORTB = 0xFD;  // Turn on blue
        do
          {
              if ((PINA & 0x40) == 0x00)  // Button B
                {
                    PORTB = 0xFF;
                    return(1);
                    }
      else if ((PINA & 0x80) == (0x00))  // Button A
                {
                    PORTB = 0xFF;
                    return(0);
                    }
      else if ((PINA & 0x20) == (0x00))  // Button C
                {
                    PORTB = 0xFF;
                    return(0);
                    }
      } while (1==1);
        }
  }
else
  {
  LCD_STRING("A)Red B)Aqua C)Green");
```

```
        if (colorsel == 2)
    {
          PORTA = 0xF0;
    PORTB = 0xFB;  // Turn on green
          do
            {
                  if ((PINA & 0x20) == 0x00)  // Button C
                    {
                          PORTB = 0xFF;
                          return(1);
                          }
      else if ((PINA & 0x80) == (0x00))  // Button A
                    {
                          PORTB = 0xFF;
                          return(0);
                          }
      else if ((PINA & 0x40) == (0x00))  // Button B
                    {
                          PORTB = 0xFF;
                          return(0);
                          }
      } while (1==1);
          }
  if (colorsel == 3)
    {
          PORTA = 0xF0;
    PORTB = 0xF9;  // Turn on blue and green
          do
            {
                  if ((PINA & 0x40) == 0x00) // Button B
                    {
        PORTB = 0xFF;
                          return(1);
                          }
      else if ((PINA & 0x80) == (0x00))  // Button A
                    {
                          PORTB = 0xFF;
                          return(0);
                          }
      else if ((PINA & 0x20) == (0x00))  // Button C
                    {
                          PORTB = 0xFF;
                          return(0);
                          }
      } while (1==1);
          }
  }
PORTB = 0xFF;
return(0);
```

```
}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
/* Original dial question
int question2()
 {
 int level = random(1,3);
 int loop = 1;
 LCD_COMMAND(0x01);
 LCD_STRING("   Turn dial to ");
 LCD_NUMBER(level);
 _delay_ms(50);                         // Wait for 500 ms
 LCD_ADDR(0x40);                        // Send cursor to address 0x40 (second row)
 _delay_ms(50);                         // Wait for 500 ms
 LCD_STRING("    'A' when set    ");
 PORTA = 0xF0;
 do // Wait for "Enter"
    {
            if ((PINA & 0x80) == 0x00) // Button A
              {
              loop = 0;
              }
            } while (loop==1);
 if (level == 1)
    {
    if ((PINC & 0xC3) == 0x80)  // Dial set to 1
       {
       return(1);
              }
    else
              {
              return(0);
              }
    }
 else if (level == 2)
    {
    if ((PINC & 0xC3) == 0xC0)  // Dial set to 2
       {
       return(1);
              }
    else
              {
              return(0);
              }
            }
 else if (level == 3)
    {
    if ((PINC & 0xC3) == 0xC2)  // Dial set to 3
       {
```

```
      return(1);
            }
   else
            {
            return(0);
            }
         }
 else
   {
   if ((PINC & 0xC3) == 0xC3)  // Dial set to 4
      {
      return(1);
            }
   else
            {
            return(0);
            }
         }
 return(0);
 }
*/

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

int question2()
 {
 int question;
 int number1;
 int answers[3];
 int correct;
 int loop;
 loop = 0;
 question = random(1,7);
 answers[0] = 0;
 answers[1] = 0;
 answers[2] = 0;
 _delay_ms(7);
 correct = random(0,2);
 if (question == 0)  // +1
   {
   number1=random(0,7);
   answers[correct] = number1+1;
         if (correct == 0)
            {
            answers[1] = answers[0]+1;
            answers[2] = answers[0]+2;
            }
   else if (correct == 1)
            {
```

```
            answers[0] = answers[1]-1;
            answers[2] = answers[1]+1;
            }
    else
            {
            answers[0] = answers[2]-2;
            answers[1] = answers[2]-1;
            }
          LCD_STRING("   What is ");
          LCD_NUMBER(number1);
          LCD_STRING(" + 1?   ");
    _delay_ms(500);                    // Wait for 500 ms
    LCD_ADDR(0x40);                          // Send cursor to address 0x40 (second row)
    _delay_ms(500);                    // Wait for 500 ms
          LCD_STRING(" A) ");
          LCD_NUMBER(answers[0]/10);
          LCD_NUMBER(answers[0]%10);
          LCD_STRING(" B) ");
          LCD_NUMBER(answers[1]/10);
          LCD_NUMBER(answers[1]%10);
    LCD_STRING(" C) ");
          LCD_NUMBER(answers[2]/10);
          LCD_NUMBER(answers[2]%10);
          LCD_CHARACTER(' ');
          }
 if (question == 1)  // +3
    {
    number1=random(0,7);
    answers[correct] = number1+3;
          if (correct == 0)
            {
            answers[1] = answers[0]+1;
            answers[2] = answers[0]+2;
            }
    else if (correct == 1)
            {
            answers[0] = answers[1]-1;
            answers[2] = answers[1]+1;
            }
    else
            {
            answers[0] = answers[2]-2;
            answers[1] = answers[2]-1;
            }
          LCD_STRING("   What is ");
          LCD_NUMBER(number1);
          LCD_STRING(" + 3?   ");
    _delay_ms(500);                    // Wait for 500 ms
    LCD_ADDR(0x40);                          // Send cursor to address 0x40 (second row)
```

```
   _delay_ms(500);                    // Wait for 500 ms
         LCD_STRING(" A) ");
         LCD_NUMBER(answers[0]/10);
         LCD_NUMBER(answers[0]%10);
         LCD_STRING(" B) ");
         LCD_NUMBER(answers[1]/10);
         LCD_NUMBER(answers[1]%10);
   LCD_STRING(" C) ");
         LCD_NUMBER(answers[2]/10);
         LCD_NUMBER(answers[2]%10);
         LCD_CHARACTER(' ');
         }
 if (question == 2)  // +5
   {
   number1=random(0,7);
   answers[correct] = number1+5;
         if (correct == 0)
           {
           answers[1] = answers[0]+1;
           answers[2] = answers[0]+2;
           }
   else if (correct == 1)
           {
           answers[0] = answers[1]-1;
           answers[2] = answers[1]+1;
           }
   else
           {
           answers[0] = answers[2]-2;
           answers[1] = answers[2]-1;
           }
         LCD_STRING("   What is ");
         LCD_NUMBER(number1);
         LCD_STRING(" + 5?   ");
   _delay_ms(500);                    // Wait for 500 ms
   LCD_ADDR(0x40);                          // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
         LCD_STRING(" A) ");
         LCD_NUMBER(answers[0]/10);
         LCD_NUMBER(answers[0]%10);
         LCD_STRING(" B) ");
         LCD_NUMBER(answers[1]/10);
         LCD_NUMBER(answers[1]%10);
   LCD_STRING(" C) ");
         LCD_NUMBER(answers[2]/10);
         LCD_NUMBER(answers[2]%10);
         LCD_CHARACTER(' ');
         }
 if (question == 3)  // +7
```

```
   {
   number1=random(0,7);
   answers[correct] = number1+7;
         if (correct == 0)
            {
            answers[1] = answers[0]+1;
            answers[2] = answers[0]+2;
            }
   else if (correct == 1)
            {
            answers[0] = answers[1]-1;
            answers[2] = answers[1]+1;
            }
   else
            {
            answers[0] = answers[2]-2;
            answers[1] = answers[2]-1;
            }
         LCD_STRING("   What is ");
         LCD_NUMBER(number1);
         LCD_STRING(" + 7?   ");
   _delay_ms(500);                        // Wait for 500 ms
   LCD_ADDR(0x40);                              // Send cursor to address 0x40 (second row)
   _delay_ms(500);                        // Wait for 500 ms
         LCD_STRING(" A) ");
         LCD_NUMBER(answers[0]/10);
         LCD_NUMBER(answers[0]%10);
         LCD_STRING(" B) ");
         LCD_NUMBER(answers[1]/10);
         LCD_NUMBER(answers[1]%10);
   LCD_STRING(" C) ");
         LCD_NUMBER(answers[2]/10);
         LCD_NUMBER(answers[2]%10);
         LCD_CHARACTER(' ');
         }
 else if (question == 4)  // Multiplication
   {
   number1=random(1,7);
   answers[correct] = number1*2;
         if (correct == 0)
            {
            answers[1] = answers[0]+2;
            answers[2] = answers[0]+4;
            }
   else if (correct == 1)
            {
            answers[0] = answers[1]-2;
            answers[2] = answers[1]+2;
            }
```

```
   else
            {
            answers[0] = answers[2]-4;
            answers[1] = answers[2]-2;
            }
         LCD_STRING("   What is ");
         LCD_NUMBER(number1);
         LCD_STRING(" * 2?   ");
   _delay_ms(500);                    // Wait for 500 ms
   LCD_ADDR(0x40);                          // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
         LCD_STRING(" A) ");
         LCD_NUMBER(answers[0]/10);
         LCD_NUMBER(answers[0]%10);
         LCD_STRING(" B) ");
         LCD_NUMBER(answers[1]/10);
         LCD_NUMBER(answers[1]%10);
   LCD_STRING(" C) ");
         LCD_NUMBER(answers[2]/10);
         LCD_NUMBER(answers[2]%10);
         LCD_STRING("  ");
         }
 else if (question == 5)  // Multiplication
   {
   number1=random(1,7);
   answers[correct] = number1*4;
         if (correct == 0)
            {
            answers[1] = answers[0]+2;
            answers[2] = answers[0]+4;
            }
   else if (correct == 1)
            {
            answers[0] = answers[1]-2;
            answers[2] = answers[1]+2;
            }
   else
            {
            answers[0] = answers[2]-4;
            answers[1] = answers[2]-2;
            }
         LCD_STRING("   What is ");
         LCD_NUMBER(number1);
         LCD_STRING(" * 4?   ");
   _delay_ms(500);                    // Wait for 500 ms
   LCD_ADDR(0x40);                          // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
         LCD_STRING(" A) ");
         LCD_NUMBER(answers[0]/10);
```

```
        LCD_NUMBER(answers[0]%10);
        LCD_STRING(" B) ");
        LCD_NUMBER(answers[1]/10);
        LCD_NUMBER(answers[1]%10);
  LCD_STRING(" C) ");
        LCD_NUMBER(answers[2]/10);
        LCD_NUMBER(answers[2]%10);
        LCD_STRING("  ");
        }
else if (question == 6)  // Multiplication
  {
  number1=random(1,7);
  answers[correct] = number1*6;
        if (correct == 0)
          {
          answers[1] = answers[0]+2;
          answers[2] = answers[0]+4;
          }
  else if (correct == 1)
          {
          answers[0] = answers[1]-2;
          answers[2] = answers[1]+2;
          }
  else
          {
          answers[0] = answers[2]-4;
          answers[1] = answers[2]-2;
          }
        LCD_STRING("   What is ");
        LCD_NUMBER(number1);
        LCD_STRING(" * 6?   ");
  _delay_ms(500);                       // Wait for 500 ms
  LCD_ADDR(0x40);                               // Send cursor to address 0x40 (second row)
  _delay_ms(500);                       // Wait for 500 ms
        LCD_STRING(" A) ");
        LCD_NUMBER(answers[0]/10);
        LCD_NUMBER(answers[0]%10);
        LCD_STRING(" B) ");
        LCD_NUMBER(answers[1]/10);
        LCD_NUMBER(answers[1]%10);
  LCD_STRING(" C) ");
        LCD_NUMBER(answers[2]/10);
        LCD_NUMBER(answers[2]%10);
        LCD_STRING("  ");
        }
else if (question == 7)  // Multiplication
  {
  number1=random(1,7);
  answers[correct] = number1*8;
```

```
        if (correct == 0)
          {
          answers[1] = answers[0]+2;
          answers[2] = answers[0]+4;
          }
  else if (correct == 1)
          {
          answers[0] = answers[1]-2;
          answers[2] = answers[1]+2;
          }
  else
          {
          answers[0] = answers[2]-4;
          answers[1] = answers[2]-2;
          }
        LCD_STRING("   What is ");
        LCD_NUMBER(number1);
        LCD_STRING(" * 8?   ");
  _delay_ms(500);                       // Wait for 500 ms
  LCD_ADDR(0x40);                               // Send cursor to address 0x40 (second row)
  _delay_ms(500);                       // Wait for 500 ms
        LCD_STRING(" A) ");
        LCD_NUMBER(answers[0]/10);
        LCD_NUMBER(answers[0]%10);
        LCD_STRING(" B) ");
        LCD_NUMBER(answers[1]/10);
        LCD_NUMBER(answers[1]%10);
  LCD_STRING(" C) ");
        LCD_NUMBER(answers[2]/10);
        LCD_NUMBER(answers[2]%10);
        LCD_STRING("  ");
        }
 if (correct==0)
  {
  do
    {
          if ((PINA & 0x80) == 0x00) // Button A
            {
                  return(1);
                  loop=1;
                  }
    else if ((PINA & 0x40) == (0x00))  // Button B
            {
            PORTB = 0xFF;
                  return(0);
                  }
    else if ((PINA & 0x20) == (0x00))  // Button C
            {
            PORTB = 0xFF;
```

```
                return(0);
                }
        } while (loop == 0);
  }
if (correct==1)
  {
  do
    {
        if ((PINA & 0x40) == 0x00) // Button B
          {
                return(1);
                loop=1;
                }
    else if ((PINA & 0x80) == (0x00))  // Button A
          {
          PORTB = 0xFF;
                return(0);
                }
    else if ((PINA & 0x20) == (0x00))  // Button C
          {
          PORTB = 0xFF;
                return(0);
                }
        } while (loop == 0);
  }
if (correct==2)
  {
  do
    {
        if ((PINA & 0x20) == 0x00)
          {
                return(1);
                loop=1;
                }
    else if ((PINA & 0x40) == (0x00))  // Button B
          {
          PORTB = 0xFF;
                return(0);
                }
    else if ((PINA & 0x80) == (0x00))  // Button A
          {
          PORTB = 0xFF;
                return(0);
                }
        } while (loop == 0);
  }
return(0);
}
```

```
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

int question3()
 {
 int question;
 int answers[3];
 int correct;
 int loop;
 loop = 0;
 question = random(0,7);
 answers[0] = 0;
 answers[1] = 0;
 answers[2] = 0;
 _delay_ms(7);
 correct = random(0,2);
 if (question == 0)  // Fibonacci
   {
         LCD_STRING("1, 1, 2, 3, 5, 8, ?");
   _delay_ms(500);                    // Wait for 500 ms
   LCD_ADDR(0x40);                          // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
   if (correct==0)
           {
           LCD_STRING(" A) 13 B) 15 C) 17  ");
           }
   else if (correct==1)
           {
           LCD_STRING(" A) 11 B) 13 C) 15  ");
           }
   else
           {
           LCD_STRING("  A) 9 B) 11 C) 13  ");
           }
         }
 else if (question == 1)  // Circuits
   {
         LCD_STRING("  Voltage is NOT:   ");
   _delay_ms(500);                    // Wait for 500 ms
   LCD_ADDR(0x40);                          // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
   if (correct==0)
           {
           LCD_STRING("A) w/i B) i*r C) p/i");
           }
   else if (correct==1)
           {
           LCD_STRING("A) p/i B) w/i C) i*r");
           }
   else
```

```
            {
            LCD_STRING("A) i*r B) p/i C) w/i");
            }
          }
else if (question == 2)  // Poetry
   {
   LCD_COMMAND(0x01);
          LCD_STRING("Who wrote The Raven?");
   _delay_ms(500);                        // Wait for 500 ms
   LCD_ADDR(0x40);                                // Send cursor to address 0x40 (second row)
   _delay_ms(500);                        // Wait for 500 ms
   if (correct==0)
            {
            LCD_STRING("A)Poe B)Donne C)Pope");
            }
   else if (correct==1)
            {
            LCD_STRING("A)Pope B)Poe C)Donne");
            }
   else
            {
            LCD_STRING("A)Donne B)Pope C)Poe");
            }
          }
else if (question == 3) // Root two
   {
   LCD_COMMAND(0x01);
          LCD_STRING("  2^(1/2) is approx ");
   _delay_ms(500);                        // Wait for 500 ms
   LCD_ADDR(0x40);                                // Send cursor to address 0x40 (second row)
   _delay_ms(500);                        // Wait for 500 ms
   if (correct==0)
            {
            LCD_STRING("A)1.41 B)1.25 C)1.67");
            }
   else if (correct==1)
            {
            LCD_STRING("A)1.67 B)1.41 C)1.25");
            }
   else
            {
            LCD_STRING("A)1.25 B)1.67 C)1.41");
            }
          }
else if (question == 4)  // Geekery
   {
   LCD_COMMAND(0x01);
          LCD_STRING(" TNG's android is:  ");
   _delay_ms(500);                        // Wait for 500 ms
```

```
   LCD_ADDR(0x40);                    // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
   if (correct==0)
            {
            LCD_STRING("A)Data B)Odo C)Troi");
            }
   else if (correct==1)
            {
            LCD_STRING("A)Troi B)Data C)Odo");
            }
   else
            {
            LCD_STRING("A)Odo B)Troi C)Data");
            }
         }
 else if (question == 5)  // Mythology
   {
   LCD_COMMAND(0x01);
   LCD_STRING("Which god is Greek? ");
   _delay_ms(500);                    // Wait for 500 ms
   LCD_ADDR(0x40);                    // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
   if (correct==0)
            {
            LCD_STRING("A)Gaia B)Loki C)Isis");
            }
   else if (correct==1)
            {
            LCD_STRING("A)Isis B)Gaia C)Loki");
            }
   else
            {
            LCD_STRING("A)Loki B)Isis C)Gaia");
            }
         }
 else if (question == 6) // Chemistry
   {
   LCD_COMMAND(0x01);
   LCD_STRING("The atomic # of C is");
   _delay_ms(500);                    // Wait for 500 ms
   LCD_ADDR(0x40);                    // Send cursor to address 0x40 (second row)
   _delay_ms(500);                    // Wait for 500 ms
   if (correct==0)
            {
            LCD_STRING("  A) 6 B) 8 C) 10   ");
            }
   else if (correct==1)
            {
            LCD_STRING("   A) 4 B) 6 C) 8   ");
```

```
            }
   else
            {
            LCD_STRING("   A) 2 B) 4 C) 6   ");
            }
         }
 else // History
   {
   LCD_COMMAND(0x01);
         LCD_STRING("D-Day was in:");
   _delay_ms(500);                        // Wait for 500 ms
   LCD_ADDR(0x40);                              // Send cursor to address 0x40 (second row)
   _delay_ms(500);                        // Wait for 500 ms
   if (correct==0)
            {
            LCD_STRING("A)1944 B)1955 C)1966");
            }
   else if (correct==1)
            {
            LCD_STRING("A)1933 B)1944 C)1955");
            }
   else
            {
            LCD_STRING("A)1922 B)1933 C)1944");
            }
         }
 if (correct==0)
   {
   do
     {
            if ((PINA & 0x80) == 0x00) // Button A
              {
                     return(1);
                     loop=1;
                     }
     else if ((PINA & 0x40) == (0x00))  // Button B
              {
              PORTB = 0xFF;
                     return(0);
                     }
     else if ((PINA & 0x20) == (0x00))  // Button C
              {
              PORTB = 0xFF;
                     return(0);
                     }
            } while (loop == 0);
   }
 if (correct==1)
   {
```

```
   do
     {
            if ((PINA & 0x40) == 0x00) // Button B
              {
                    return(1);
                    loop=1;
                    }
     else if ((PINA & 0x80) == (0x00))  // Button A
              {
              PORTB = 0xFF;
                    return(0);
                    }
     else if ((PINA & 0x20) == (0x00))  // Button C
              {
              PORTB = 0xFF;
                    return(0);
                    }
            } while (loop == 0);
   }
 if (correct==2)
   {
   do
     {
            if ((PINA & 0x20) == 0x00)
              {
                    return(1);
                    loop=1;
                    }
     else if ((PINA & 0x40) == (0x00))  // Button B
              {
              PORTB = 0xFF;
                    return(0);
                    }
     else if ((PINA & 0x80) == (0x00))  // Button A
              {
              PORTB = 0xFF;
                    return(0);
                    }
            } while (loop == 0);
   }
 return(0);
 }

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

int main()
{
 sec=0;
 hour=0;
```

```
min=0;
ahour=0;
amin=0;
aset=0;
alarmactive=0;
alarmsel=0;
int wait=0;
int q1=0;
int q2=0;
int q3=0;
int i;

TIMSK = 0x10;
TCCR1B = 0x0C;
OCR1A = 3906;  // Interrupt every second

DDRA = 0x0F;  // 4 inputs, 4 outputs
DDRB = 0xFF;  // All outputs
DDRC = 0x00;  // All inputs
DDRD = 0xFF;  // All outputs

LCD_INIT();

updateLCD(min, hour, amin, ahour, aset);
PORTB = 0xFF;   // LED turned off
PORTA = 0x04;   // Reset CPLD display to 00:00
_delay_ms(10);
PORTA = 0x00;
sei();
do // Main loop runs forever
  {
        PORTA = 0xF0;
  _delay_ms(50);                        // Wait for 500 ms
  LCD_ADDR(0x40);                               // Send cursor to address 0x40 (second row)
  _delay_ms(50);                        // Wait for 500 ms
        LCD_STRING("  Press 'A' to set  ");
        _delay_ms(500);
        _delay_ms(500);
        // User sets Alarm or Clock Time
  if ((PINA & 0x80) == 0x00)  // Button A/"Set" pressed
          {
    _delay_ms(50);                      // Wait for 500 ms
    LCD_ADDR(0x40);                               // Send cursor to address 0x40 (second row)
    _delay_ms(50);                      // Wait for 500 ms
          LCD_STRING("B) Alarm or C) Time?");
          wait = 1;
          do
            {
                // Alarm
```

```
        if ((PINA & 0x40) == 0x00) // Button B/"Alarm"
          {
                do
                  {
_delay_ms(50);                          // Wait for 500 ms
LCD_ADDR(0x40);                                 // Send cursor to address 0x40 (second row)
_delay_ms(50);                          // Wait for 500 ms
      LCD_STRING("A) None B) Hr C) Min");
            _delay_ms(500);
                  if ((PINA & 0x80) == 0x00) // Button A/"Enter"
                    {
                    aset = 0;               // No alarm stored
                          wait = 0;               // Break loops
                          }
else if (((PINA & 0x40) == 0x00) || ((PINA & 0x20) == 0x00))  // If Button B or C
                          {
                          do
                      {
                            PORTA = 0xF0;
                                  if ((PINA & 0x80) == 0x00) // Button A/"Enter"
                                    {
                                          wait = 0;           // Break loops
                                          aset = 1;       // An alarm is stored
                                          }
    else if ((PINA & 0x40) == 0x00) // Button B/"Hour"
                                    {
                                          aset = 1;
                                          if (ahour < 23)
          {
                                                ahour+=1;           // Alarm set one hour ahead
                                                }
        else
                                            {
          ahour = 0;
                                                }
                                          updateLCD(min, hour, amin, ahour, aset);
                                          _delay_ms(500);
                                          }
    else if ((PINA & 0x20) == 0x00) // Button C/"Minute"
                                    {
                                          aset = 1;
                                          if (amin < 59)
                                            {
        amin+=1;          // Alarm set one minute ahead
                                                }
        else
                                            {
                                                amin = 0;
                                                }
```

```
                updateLCD(min, hour, amin, ahour, aset);
            _delay_ms(500);
                }
        _delay_ms(50);                  // Wait for 500 ms
    LCD_ADDR(0x40);                 // Send cursor to address 0x40 (second row)
    _delay_ms(50);              // Wait for 500 ms
        LCD_STRING("A) Done B) Hr C) Min");
    } while (wait==1);
            }
} while (wait==1);
        }
    if ((PINA & 0x20) == 0x00) // Button C/"Time"
      {
        do
          {
            _delay_ms(500);
            PORTA = 0xF0;
      if ((PINA & 0x80) == 0x00) // Button A/"Enter"
            {
            wait = 0;
            }
else if ((PINA & 0x40) == 0x00) // Button B/"Hour"
  {
            if (hour < 23)
              {
                hour+=1;
                }
  else
              {
                hour = 0;
                }
            updateLCD(min, hour, amin, ahour, aset);
        /*  for (int i=0; i<60; i++)
          {
              PORTA = 0x02;  // 60 pulses to CPLD to update display
              } */
          _delay_ms(500);
  }
else if ((PINA & 0x20) == 0x00) // Button C/"Minute"
            {
  if (min < 59)
              {
              min+=1;
    }
  else
              {
                min = 0;
                }
            updateLCD(min, hour, amin, ahour, aset);
```

```
                    //  PORTA = 0x02;  // 1 pulse to CPLD to update display
                _delay_ms(500);
                    }
                _delay_ms(50);                    // Wait for 500 ms
        LCD_ADDR(0x40);                    // Send cursor to address 0x40 (second row)
        _delay_ms(50);                    // Wait for 500 ms
            LCD_STRING("A) Done B) Hr C) Min");
        } while (wait==1);
            }
    } while (wait==1);
  }

// Alarm Trigger
      if ((aset==1)&&(min==amin)&&(hour==ahour)) // disarming
  {
        alarmactive = 1;
        alarmsel = random(0,5);   // choose a random number for the alarm
  LCD_COMMAND(0x01);
        _delay_ms(10);
        alarmactive=1;
        do
          {
    q1 = question1();
    if (q1 == 1)
            {
                    q1 = 0;
                    LCD_COMMAND(0x01);
            LCD_STRING("  Question 1 of 4   ");
      _delay_ms(50);                    // Wait for 50 ms
      LCD_ADDR(0x40);                    // Send cursor to address 0x40 (second row)
      _delay_ms(50);                    // Wait for 50 ms
                    LCD_STRING("      Correct!      ");
      for (i = 0; i <4; i++)
                    {
                _delay_ms(500);
                }
      LCD_COMMAND(0x01);
            q2 = question2();
            if (q2 == 1)
          {
                _delay_ms(50);                    // Wait for 50 ms
                    LCD_COMMAND(0x01);
          LCD_STRING("  Question 2 of 4   ");
        _delay_ms(50);                    // Wait for 50 ms
        LCD_ADDR(0x40);                    // Send cursor to address 0x40 (second row)
        _delay_ms(50);                    // Wait for 50 ms
                    LCD_STRING("      Correct!      ");
        for (i = 0; i <4; i++)
                    {
```

```
                    _delay_ms(500);
                    }
        LCD_COMMAND(0x01);
                q3 = question3();
        if (q3 == 1)
                {
                    q3 = 0;
                LCD_COMMAND(0x01);
            LCD_STRING("  Question 3 of 4   ");
          _delay_ms(50);                // Wait for 50 ms
          LCD_ADDR(0x40);                   // Send cursor to address 0x40 (second row)
          _delay_ms(50);                // Wait for 50 ms
                LCD_STRING("      Correct!      ");
          for (i = 0; i <4; i++)
                {
                  _delay_ms(500);
                  }
          LCD_COMMAND(0x01);
                    q3 = question3();
                    if (q3 == 1)
                      {
                          q3 = 0;
                          TIMSK = 0x10;
                          TCCR0 = 0x00;
                          alarmactive=0;
                          }
                    }
        }
      }
    if (alarmactive == 1)
            {
                LCD_COMMAND(0x01);
            updateLCD(min, hour, amin, ahour, aset);
      _delay_ms(50);                // Wait for 50 ms
      LCD_ADDR(0x40);               // Send cursor to address 0x40 (second row)
      _delay_ms(50);                // Wait for 50 ms
                LCD_STRING("    Incorrect     ");
      for (i = 0; i < 20; i++)
            {
        _delay_ms(500);
                  }
                }
    } while (alarmactive == 1);
  }
 } while (1==1);
return(0);
}
```

Resources:

Alexander, Charles K., and Matthew N. Sadiku. Fundamentals of Electric Circuits. New York: McGraw-Hill Companies, The, 2005.

All Electronics | Electronic and Electro-Mechanical Parts and Supplies. <http://www.allelectronics.com>.

"ATmega32 manual." Atmel Corporation - Industry Leader in the Design and Manufacture of Advanced Semiconductors. <http://www.atmel.com/dyn/resources/prod_documents/doc2503.pdf>.

"Avr-libc: : Interrupts." Savannah. <http://www.nongnu.org/avr-libc/user-manual/group__avr__interrupts.html>.

"Ce2800:timers [TaylorWiki]." Main [TaylorWiki]. <http://wiki.msoe.us/ce2800/timers>.

"Climber's info on AVR programming - Interrupts." Shaw Webspace. <http://members.shaw.ca/climber/avrinterrupts.html>.

EEL 4914C. <http://www.bosman.ece.ufl.edu/EEL3923CJDF08/EEL3923CJDClassSyllabusF08.html>.

EEL 4924: Electrical Engineering Design (Senior Design). University of Florida. <http://mil.ufl.edu/4924/>.

EEL4744 - Microprocessor Applications. University of Florida. <http://www.add.ece.ufl.edu/4744/>.

"PWM - An Overview." <http://www.societyofrobots.com/member_tutorials/book/export/html/228>.

Savitch, Walter J. Absolute C++. New York: Addison-Wesley Longman, Incorporated, 2005.