Drexel University

To: Dr. Peters

From: Tyler Ostinato

Date: October 3, 2021

Re: ECE 303-Lab 2-Timers and Interrupts

Purpose

Create a game of “Code Breaker” utilizing timers and interrupts on the Arduino Mega 2560. This game should take in a player’s input and compare it against a randomly generated number. If the player does not guess all 4 integers before round 5 the game is over.

Discussion

There were two main parts to this lab, the circuit design, and the coding implementation of “Code Breaker”.

Breadboard Circuit | 1.4

In this section of the lab required wiring the physical setup of the game codebreaker. First I wired all 4 LEDs in individual series circuits attached to a 1kΩ resistor(Fig. 1). Once task for implementing the LEDs was ensuring each one was attached to a separate timer on the Arduino (Fig. 2). To do this I plug LED 1,2,3,4 into digital pins 44,6,5,11 respectively. This allowed me to change the flashing rate of each LED individually based on a player’s guess.

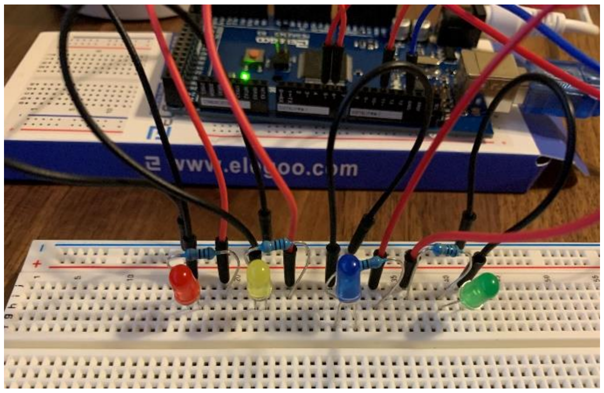


Figure 1: 4 LEDs wired with 1kΩ resistors.

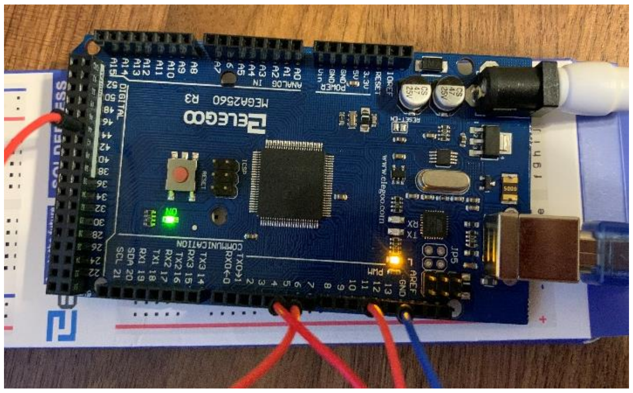


Figure 2: Digital pins connecting LEDs to timers 1,3,4, and 5.

Arduino Sketch | 1.5

The Arduino sketch for this lab was quite a bit more difficult than the breadboard. First step was implementing a timer using a compare vector. What this allows us to do is change the flashing rate of the LED by changing how fast our timer hits a reset flag using register OCRnx (Fig. 3).

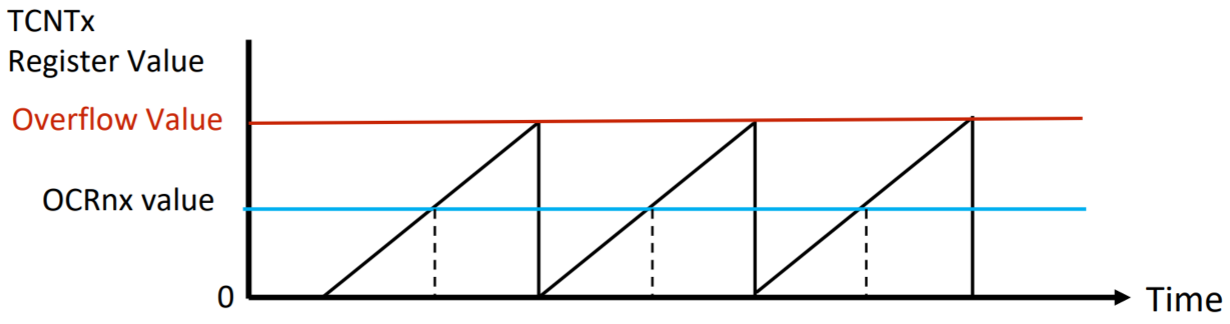
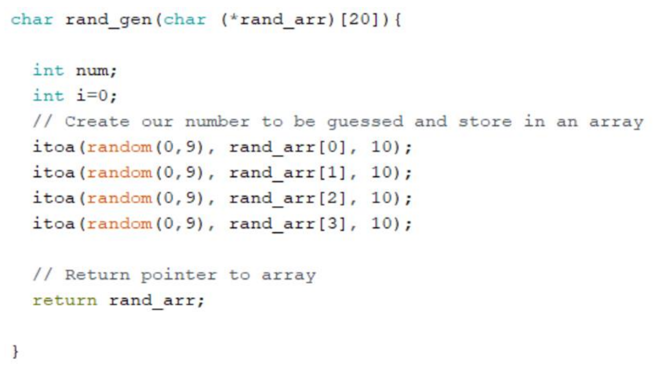


Figure 1: Graphical representation of using the compare register on a timer.

After implementing the timers, the rest of the code involved the "Code Breaker" game logic. My first task to create the game logic was creating a randomly generated number. I created a random integer using random(0,9) and storing each number into an array (Fig. 4). Next was reading a player's input via the serial terminal and storing that answer into a separate array (Fig. 5). Once both numbers have been received, conditional statements are used to compare the values. If a value is the same the corresponding LED is turned off, otherwise the corresponding LED blinks faster.



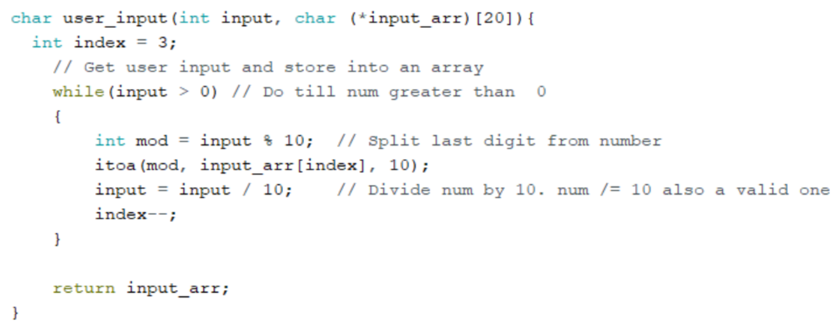
```
char rand_gen(char (*rand_arr)[20]){

  int num;
  int i=0;
  // Create our number to be guessed and store in an array
  itoa(random(0,9), rand_arr[0], 10);
  itoa(random(0,9), rand_arr[1], 10);
  itoa(random(0,9), rand_arr[2], 10);
  itoa(random(0,9), rand_arr[3], 10);

  // Return pointer to array
  return rand_arr;

}
```

Figure 4: Code to create a random integer for the player to guess.

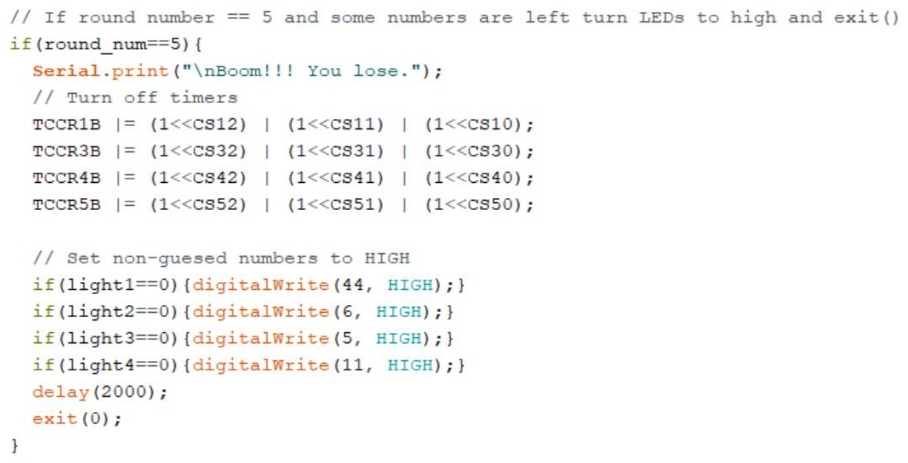


```
char user_input(int input, char (*input_arr)[20]){
  int index = 3;
    // Get user input and store into an array
    while(input > 0) // Do till num greater than  0
    {
        int mod = input % 10;  // Split last digit from number
        itoa(mod, input_arr[index], 10);
        input = input / 10;    // Divide num by 10. num /= 10 also a valid one
        index--;
    }

    return input_arr;
}
```

Figure 5: Code to take a player's guess and parse it into an array.

The last step was to end the game when a player hits round 5, or if they guess all the correct digits in the code. To handle the end of the game I used a counter to get the current round number. Once the counter hit 5 the remaining LED's timers would be turned off and the LEDs would be set to HIGH indicating to the player which numbers were not guessed in time.



```
// If round number == 5 and some numbers are left turn LEDs to high and exit()
if(round_num==5){
  Serial.print("\nBoom!!! You lose.");
  // Turn off timers
  TCCR1B |= (1<<CS12) | (1<<CS11) | (1<<CS10);
  TCCR3B |= (1<<CS32) | (1<<CS31) | (1<<CS30);
  TCCR4B |= (1<<CS42) | (1<<CS41) | (1<<CS40);
  TCCR5B |= (1<<CS52) | (1<<CS51) | (1<<CS50);

  // Set non-guesed numbers to HIGH
  if(light1==0){digitalWrite(44, HIGH);}
  if(light2==0){digitalWrite(6, HIGH);}
  if(light3==0){digitalWrite(5, HIGH);}
  if(light4==0){digitalWrite(11, HIGH);}
  delay(2000);
  exit(0);
}
```

Figure 6: Code to handle reaching round 5 without break the code.

Conclusion

In this lab I got a great deal of experience with interrupts and timers on the Arduino Mega 2560. This will prove helpful in future labs when I need to invoke the Arduino hardware when writing a sketch.

Appendix

Attached is the sketch used for my lab 2.

```
int input;
int game_flag = 0;
int round_num = 1;
int correct_guess = 0;
char input_arr[4][20];
char rand_arr[4][20];
long int compare_val1 = 49984;
long int compare_val2 = 49984;
long int compare_val3 = 49984;
long int compare_val4 = 49984;
int light1=0;
int light2=0;
int light3=0;
int light4=0;

//int get_next_timing

void setup() {
  // Setup serial terminal
  Serial.begin(9600);
  randomSeed(analogRead(0));

  // put your setup code here, to run once:
  pinMode(5, OUTPUT);
  digitalWrite(5, LOW);
```

```
pinMode(11, OUTPUT);

digitalWrite(11, LOW);


pinMode(6, OUTPUT);

digitalWrite(6, LOW);


pinMode(44, OUTPUT);

digitalWrite(44, LOW);


noInterrupts();

// Timer for PIN 5 (Timer 3)

TCCR3A=0;

TCCR3B=0;

TIMSK3=0;

TCNT3=0;

OCR3A=49984;

TCCR3B |=(1<<WGM32); // Waveform gen mode 4

TCCR3B |= (0<<CS32) | (1<<CS31) | (1<<CS30); // 1024 prescaler

TIMSK3 |=(1<<OCIE3A); // Enable timer 3 pin 5 output compare flag


// Timer for PIN 11 (Timer 1)

TCCR1A=0;

TCCR1B=0;

TIMSK1=0;

TCNT1=0;

OCR1A=49984;

TCCR1B |=(1<<WGM12); // Waveform gen mode 4

TCCR1B |= (0<<CS12) | (1<<CS11) | (1<<CS10); // 1024 prescaler

TIMSK1 |=(1<<OCIE1A);


// Timer for PIN 6 (Timer 4)

TCCR4A=0;

TCCR4B=0;

TIMSK4=0;
```

```
  TCNT4=0;

  OCR4A=49984;

  TCCR4B |=(1<<WGM42); // Waveform gen mode 4

  TCCR4B |= (0<<CS42) | (1<<CS41) | (1<<CS40); // 1024 prescaler

  TIMSK4 |=(1<<OCIE4A);


  // Timer for PIN 44 (Timer 5)

  TCCR5A=0;

  TCCR5B=0;

  TIMSK5=0;

  TCNT5=0;

  OCR5A=49984;

  TCCR5B |=(1<<WGM52); // Waveform gen mode 4

  TCCR5B |= (0<<CS52) | (1<<CS51) | (1<<CS50); // 10246 prescaler

  TIMSK5 |=(1<<OCIE5A);


  interrupts();

}



char user_input(int input, char (*input_arr)[20]){

  int index = 3;

    // Get user input and store into an array

    while(input > 0) // Do till num greater than  0

    {

        int mod = input % 10;  // Split last digit from number

        itoa(mod, input_arr[index], 10);

        input = input / 10;    // Divide num by 10. num /= 10 also a valid one

        index--;

    }


    return input_arr;

}
```

```
char rand_gen(char (*rand_arr)[20]){

  int num;
  int i=0;
  // Create our number to be guessed and store in an array
  itoa(random(0,9), rand_arr[0], 10);
  itoa(random(0,9), rand_arr[1], 10);
  itoa(random(0,9), rand_arr[2], 10);
  itoa(random(0,9), rand_arr[3], 10);

  // Return pointer to array
  return rand_arr;

}



char compare_logic(char (*input_arr)[20], char (*rand_arr)[20]){
  // Get input and random number and compare them
  // Figure out logic for just the first # then do all four

  Serial.print("Code: ");
  Serial.print(rand_arr[0]);Serial.print(rand_arr[1]);Serial.print(rand_arr[2]);Serial.print(rand_arr[3]);
  Serial.print("\n");
  Serial.print("Your guess: ");
  Serial.print(input_arr[0]);Serial.print(input_arr[1]);Serial.print(input_arr[2]);Serial.print(input_arr[3]);
  Serial.print("\n");

  // Timer 5
  if( atoi(input_arr[0]) == atoi(rand_arr[0]) && input_arr[0] != "|"){
    strcpy(rand_arr[0], "|");
    correct_guess++;
    TCCR5B |= (1<<CS52) | (1<<CS51) | (1<<CS50);
    digitalWrite(44, LOW);
    light1=1;
  }
```

```
else{
  // Blink Faster
  OCR5A=compare_val1;
  compare_val1=compare_val1/2;
}
// Timer 4
if( atoi(input_arr[1]) == atoi(rand_arr[1]) && input_arr[1] != "|"){
strcpy(rand_arr[1], "|");
correct_guess++;
TCCR4B |= (1<<CS42) | (1<<CS41) | (1<<CS40);
digitalWrite(6, LOW);
light2=1;
}
else{
  OCR4A=compare_val2;
  compare_val2=compare_val2/2;
}
// Timer 3
if( atoi(input_arr[2]) == atoi(rand_arr[2]) && input_arr[2] != "|"){
  strcpy(rand_arr[2], "|");
  correct_guess++;
  TCCR3B |= (1<<CS32) | (1<<CS31) | (1<<CS30);
  digitalWrite(5, LOW);
  light3=1;
}
else{
  // Blink faster
  OCR3A=compare_val3;
  compare_val3=compare_val3/2;
}
// Timer 1
if( atoi(input_arr[3]) == atoi(rand_arr[3]) && input_arr[3] != "|"){
  strcpy(rand_arr[3], "|");
  correct_guess++;
  TCCR1B |= (1<<CS12) | (1<<CS11) | (1<<CS10);
```

```
    digitalWrite(11, LOW);
    light4=1;
  }
  else{
    // Blink faster
    OCR1A=compare_val4;
    compare_val4=compare_val4/2;
  }

  Serial.print("Not yet guessed: ");
  Serial.print(rand_arr[0]);Serial.print(rand_arr[1]);Serial.print(rand_arr[2]);Serial.print(rand_arr[3]);
  Serial.print("\n\n----------------------------------------\n");

  // Return the updated array
  return rand_arr;
}

void loop() {

  // put your main code here, to run repeatedly:
  if (Serial.available() >= 0) {
    input = Serial.parseInt();

    if(input){
      // Get user input
      user_input(input, input_arr);
      // Create random number to guess
      if(game_flag==0){
        Serial.print("\n----------------------------------------\n");
        rand_gen(rand_arr);
        game_flag=1;
      }

      Serial.print("\n");
      Serial.print("Starting round ");
    Serial.print(round_num);

    Serial.print("\n");


    // Compare user input to random number

    compare_logic(input_arr, rand_arr);


    // End game when if code has been guessed

    if(correct_guess==4){

     Serial.print("\nCongrats you've won!!!");

     delay(2000);

     exit(0);

    }


    // If round number == 5 and some numbers are left turn LEDs to high and exit()

    if(round_num==5){

     Serial.print("\nBoom!!! You lose.");

     // Turn off timers

     TCCR1B |= (1<<CS12) | (1<<CS11) | (1<<CS10);

     TCCR3B |= (1<<CS32) | (1<<CS31) | (1<<CS30);

     TCCR4B |= (1<<CS42) | (1<<CS41) | (1<<CS40);

     TCCR5B |= (1<<CS52) | (1<<CS51) | (1<<CS50);


     // Set non-guesed numbers to HIGH

     if(light1==0){digitalWrite(44, HIGH);}

     if(light2==0){digitalWrite(6, HIGH);}

     if(light3==0){digitalWrite(5, HIGH);}

     if(light4==0){digitalWrite(11, HIGH);}

     delay(2000);

     exit(0);

    }

    round_num++;


   }

  }

 }
```

```
ISR(TIMER3_COMPA_vect){
  digitalWrite(5, !digitalRead(5));
}

ISR(TIMER1_COMPA_vect){
  digitalWrite(11, !digitalRead(11));
}

ISR(TIMER4_COMPA_vect){
  digitalWrite(6, !digitalRead(6));
}

ISR(TIMER5_COMPA_vect){
  digitalWrite(44, !digitalRead(44));
}
```