Before accessing the port directly, there were observable amount of inaccuracy, but when putting in the PORTB code the wavelengths were more precise and consistent. I noticed that the time it took for the voltage to change from low to high was less when accessing it directly. Part 3 is much faster because of the direct access to the port, and no need to access all the extra functions.

Here is my code and screenshots for Lab 2:

```
// Terran Blake / Lab 2 / 7:30am Wednesday / ECE 241

// Variables will change :
int LedTimer = LOW;            // Integer used to set up LED
int shortTimer = LOW;
unsigned long oldMillis = 0;       // will store last time LED was updated

// constants won't change :
const long interval = 20;          // How often LED will blink
const long shortinterval = 10;

void setup()
{
  // Makes pin 13 as an OUTPUT
  pinMode(13, OUTPUT);
  pinMode(10, OUTPUT);
}

void loop()
{

  // Will use the timer to find whether it is time to blink or not
  unsigned long currentMillis = millis();

  if(currentMillis - oldMillis >= interval)
  {
    // save the last time you blinked the LED
    oldMillis = currentMillis;

    // if the LED is off turn it on and vice-versa:
    if (LedTimer == LOW)
      LedTimer = HIGH;
       PORTB |= 0x10; // Force bit 2, pin 12, high
    else
      LedTimer = LOW;

    // set the LED with the ledState of the variable:
    digitalWrite(13, LedTimer);
  PORTB &= ~0x10; // Force bit 2, Pin 12, low

  //Same idea as the last counter but on a much faster pace and must be tested with Logic Probe
```

```
 if(currentMillis - oldMillis >= shortinterval)
 {
  oldMillis = currentMillis;

  if (shortTimer == LOW)
  {
   shortTimer = HIGH;
   PORTB |= 0x10; // Force bit 2, pin 12, high
  }
  else
  {
   shortTimer = LOW;
  }
   //Assigns timer to pin
   digitalWrite(12, shortTimer);
   PORTB &= ~0x10; // Force bit 2, Pin 12, low
  }
 } //end of loop
} //end of program
```