Sistemas Microcontrolados
Experimento 06 - Base de Tempo

Aluno: Deivid da Silva Galvão RA: 2408740
Aluno: Eduardo Yuji Yoshida Yamada RA: 2320606

1 Introdução

Neste experimento, criamos um programa que transmite horas, minutos e segundos para a interface serial do arduino utilizando o comando de interrupção por TIMER via registradores.

2 Materiais e Métodos

2.1 Materiais Utilizados

- 3 LEDs brancos;
- 3 resistências de 220 Ω;
- Jumpers (fios para conexão);
- Placa Arduino Mega 2560;
- Protoboard para montagem do circuito.
- Potenciômetro

2.2 Esquemático do Circuito

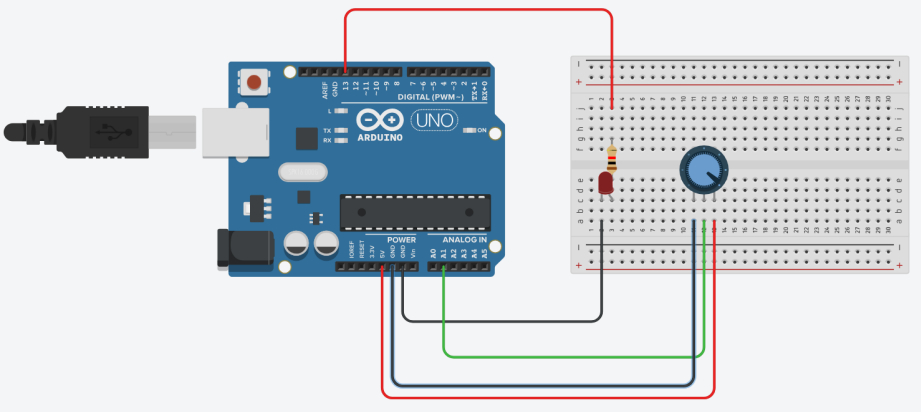

Figure 1: Circuito montado no Tinkercad

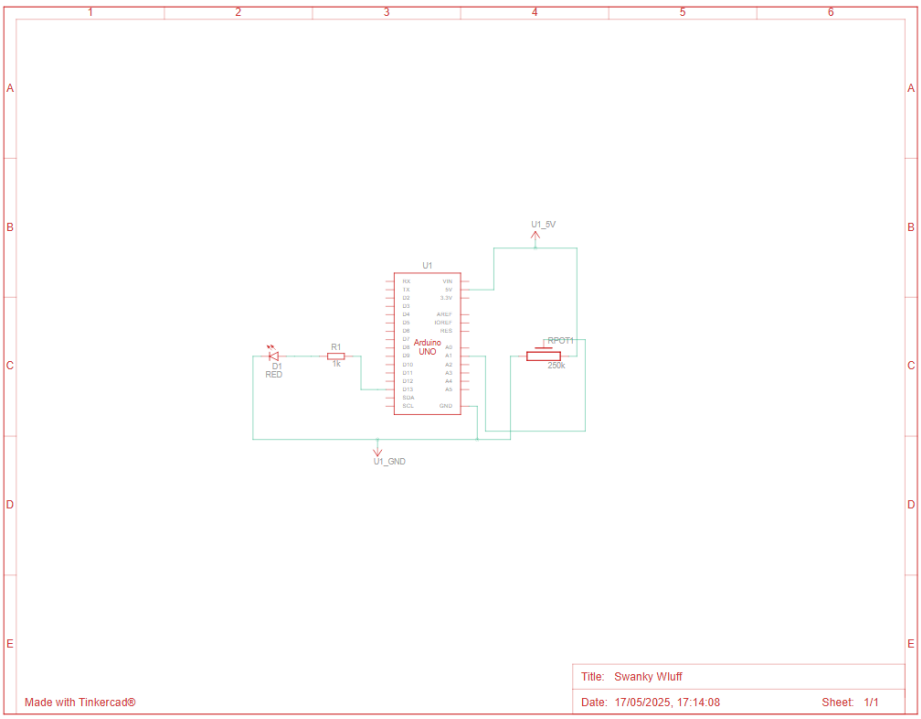

Figure 2: Diagrama esquemático do circuito no Tinkercad

2.3 Código Fonte Comentado

```
#define ledPin 13  // LED conectado no pino 13
int analogPin = A1;  // Pino A1 usado para o potenciometro
```

```
int count, led_pisca, val, pot_ts, dezena, unidade, valor;
int comunicacao_ts, relogio_ts, horas, minutos, segundos;

void setup() {
  pinMode(ledPin, OUTPUT);  // Define o pino do LED como saida
  Serial.begin(2000000);    // Inicia comunicacao serial

  // Configuracoes do TIMER1
  TCCR1A = 0;
  TCCR1B = 0;
  TCCR1B |= (1 << CS10);    // Usa clock sem prescaler
  TCNT1 = 0xC180;           // Valor inicial do timer
  TIMSK1 |= (1 << TOIE1);   // Habilita interrupcao de
     overflow do TIMER1
}

void loop() {
  // Controle do LED com efeito de fade
  if (count) {
    count = 0;
    led_pisca++;
    if (led_pisca > 600) {
      led_pisca = 0;
    }
    if (led_pisca < 300) {
      analogWrite(ledPin, map(led_pisca, 0, 300, 0, 255));
    } else {
      analogWrite(ledPin, map(led_pisca, 300, 600, 255, 0));
    }
  }

  // Leitura do potenciometro a cada intervalo
  if (pot_ts > 10) {
    pot_ts = 0;
    val = analogRead(analogPin);
    valor = map(analogRead(analogPin), 0, 1023, 0, 99);
    dezena = valor / 10;
    unidade = valor % 10;
  }

  // Incrementa tempo (tipo um relogio simples)
  if (relogio_ts > 1000) {
    relogio_ts = 0;
    segundos++;
    if (segundos > 59) {
      segundos = 0;
      minutos++;
```

```
      if (minutos > 59) {
        minutos = 0;
        horas++;
      }
    }
  }

  // Envia dados pela serial periodicamente
  if (comunicacao_ts > 100) {
    comunicacao_ts = 0;
    Serial.print("Potenciometro:");
    Serial.println(val);
    Serial.print("Horas:");
    Serial.println(horas);
    Serial.print("Minutos:");
    Serial.println(minutos);
    Serial.print("Segundos:");
    Serial.println(segundos);
  }
}

// Interrupcao do TIMER1, dispara periodicamente
ISR(TIMER1_OVF_vect) {
  TCNT1 = 0xC180;  // Reinicia o TIMER
  count++;
  pot_ts++;
  relogio_ts++;
  comunicacao_ts++;
}
```

Listing 1: Código para controle PWM dos LEDs

3 Resultados

O sistema implementado mostrou o LED piscando corretamente, aumentando sua luminosidade ao longo de 300 ms e diminuindo ao longo de 300 ms através do TIMER configurado.

O potenciômetro também trabalhou de forma correta, mostrando os valores de 0 até 1023.

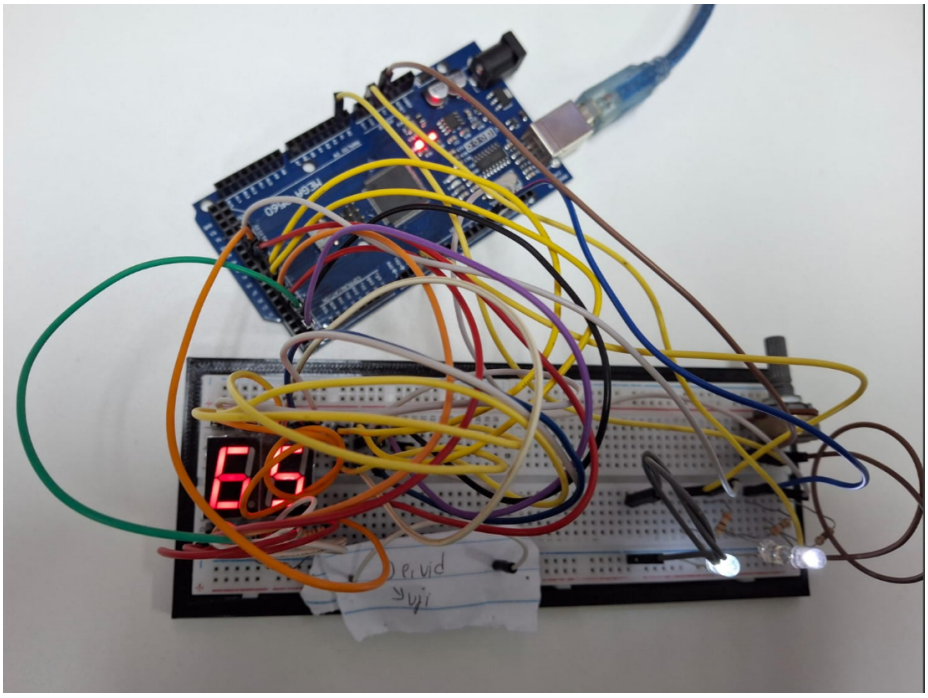

Figure 3: Montagem física do circuito no laboratório

4 Conclusão

Por fim, pode-se concluir que o experimento demonstrou de forma prática a aplicação das interrupções por *TIMER* via registradores para a criação de uma base de tempo precisa e para a transmissão eficiente de horas, minutos e segundos através da interface serial do Arduino. Ao configurar o *TIMER1* e utilizar sua interrupção, foi possível realizar a contagem dos intervalos temporais e, simultaneamente, coordenar outras tarefas, como o controle do *PWM* dos *LEDs* e a leitura do potenciômetro, sem que o sistema ficasse bloqueado durante sua execução. Essa abordagem ressalta a importância da integração entre técnicas de controle temporal e comunicação serial em sistemas microcontrolados.