Ministério da Educação
Universidade Tecnológica Federal do Paraná
Coordenação de Engenharia de Computação
Materiais e Equipamentos Elétricos
Campus Apucarana

Laboratório de Materiais e Equipamentos Elétricos

Roteiro 8: Arduino - Timer e Entrada Digital

1. Equipe

Nome	RA
Gabriel Felipe Ferdinandi de Souza	2669480
Gustavo Henrique Gonçalves	2669528
Gustavo Ferreira Fonseca	2669510

2. Desenvolvimento da prática

Código A:

```
//Declarando variaveis:
#define led1 13
#define led2 12
unsigned long tempo_anterior = 0, tempo_atual = 0;
bool estado = 0, estado2 = 0;
unsigned long tempo_anterior2 = 0, tempo_atual2 = 0;

void setup(){

  //Setando as pinagens:
  pinMode(led2, OUTPUT);
  pinMode(led1, OUTPUT);
```

```
} //Fim SETUP

void loop(){
  //Atualizando tempo atual com a função millis
  tempo_atual = millis();
  tempo_atual2 = millis();

  //Duas verificações if para criar as alterações assincronas:

  if ((tempo_atual - tempo_anterior) > 1000){

    estado =! estado;
    digitalWrite(led1, estado);
    tempo_anterior = tempo_atual;

    } //Fim if

  if ((tempo_atual2 - tempo_anterior2) > 700){

    estado2 =! estado2;
    digitalWrite(led2, estado2);
    tempo_anterior2 = tempo_atual2;

    } //Fim if

} //Fim LOOP
```

Código B:

```
//Declarando as variaveis:
#define LEDG 12
#define LEDR 13
#define BUTTON 2
bool estado_vermelho = 0, estado_button = 0, estado_verde = 0,
estado_button_anterior = 0;
unsigned long tempo_atual = 0, tempo_anterior = 0;

void setup(){

  //Definindo as pinagens:
```

```
  pinMode(LEDG, OUTPUT);
  pinMode(LEDR, OUTPUT);
  pinMode(BUTTON, INPUT);

} //Fim SETUP

void desliga(){ //Função para desligar o led vermelho

    estado_vermelho = 0; //Puxa para LOW o estado_vermelho
    digitalWrite(LEDR, estado_vermelho);

} //Fim DESLIGA

void led(){ //Função para trocar estado led vermelho pelo botão

  estado_button = digitalRead(BUTTON);

  if(estado_button == 1 && estado_button_anterior == 0){

    estado_vermelho =! estado_vermelho;
    digitalWrite(LEDR, estado_vermelho);

  } //Fim if

  estado_button_anterior = estado_button;

} //Fim LED



void loop(){

  tempo_atual = millis();
  estado_button = digitalRead(BUTTON); //Recebe o estado atual do
botão

  if((tempo_atual - tempo_anterior) > 1000){

    estado_verde =! estado_verde;
    digitalWrite(LEDG, estado_verde);
    desliga(); //Chama a função desliga a cada troca do estado do led
verde
```

```
    tempo_anterior = tempo_atual;

  } //Fim if

  attachInterrupt(digitalPinToInterrupt(BUTTON), led, CHANGE); //A
cada troca de estado do botão chama a função do botão

} //Fim LOOP
```

Código C:

```
//Declarando as variaveis:
#define LEDG 13
#define BUTTON 2
bool estado_button = 0, estado_button_anterior = 0, estado_verde = 0;
unsigned long tempo_atual = 0, tempo_anterior = 0;

void setup(){

  //Definindo as pinagens:
  pinMode(LEDG, OUTPUT);
  pinMode(BUTTON, INPUT);

} //Fim SETUP


void loop(){

  tempo_atual = millis();
  estado_button = digitalRead(BUTTON); //Recebe o estado atual do
botão

  if(estado_button == 1 && estado_button_anterior == 0 && (tempo_atual
- tempo_anterior) > 200){ //Coreção do efeito bounce pelo aguardo de
200 mS

    estado_verde =! estado_verde;
    digitalWrite(LEDG, estado_verde);
```

```
    tempo_anterior = tempo_atual;

  } //Fim if

  estado_button_anterior = estado_button;

} //Fim VOID
```

Código D:

PROFESSOR, CASO QUEIRA TESTAR OU SUGERIR MELHORIAS DO CÓDIGO “D”, ENTRE EM CONTATO QUE LHE ENVIO O ARQUIVO .INO!

```
/*
=====================================
JOGO DA TOUPEIRA:
DESENVOLVIDO POR: Gabriel Felipe F. de Souza
PROFESSOR RESPONSÁVEL: Prof. Dr. Daniel Prado de Campos
DATA: 09/05/2024
VERSÃO: 1.0
-------------------------------
OBSERVAÇÕES: Neste jogo o objetivo é clicar no botão enquanto o led
verde está ativo, caso clique
antes, o led vermelho acende por 2 segundos e reseta o jogo, caso
clique na marca, o led vermelho pisca
por 10 vezes de 50 milisSegundos, e caso você não faça nada, o led
vermelho pisca 3 vezes assíncrono de
100-500 milisSegundos, caso você segure o botão antes do led verde
acender, e solte-o na sua ligação,
o jogo desconsidera este "clique".
-------------------------------
FUTURAS MELHORIAS POSSÍVEIS: Pode ser testado outros métodos de
compressão de código, a fim de deixá-lo
menor e mais  otimizado, além disso não fiz o uso da interrupção e
poderia implementá-lo em alguma
parte do jogo, no quesito de redução de código!
=====================================
*/
```

```
//Declarando as variáveis:
#define LEDG 13
#define LEDR 12
#define BUTTON 2
bool estado_button = 0, estado_button_anterior = 0, estado_verde = 0,
estado_vermelho = 0;
//Foi separado em 2 tempos de timers para gerar randoms diferentes
para ligar o led verde e para apagá-lo:
unsigned long tempo_atual = 0, tempo_anterior = 0, tempo_topeira = 0,
tempo_anterior2 = 0 ,tempo_topeira2 = 0;

void reset(){ //Função que após ser chamada reseta os parametros do
game e atribui novos valores random

    tempo_atual = millis();

    tempo_anterior = tempo_atual;
    tempo_anterior2 = tempo_atual;
    tempo_topeira = random(400, 1500);
    tempo_topeira2 = random(400, 1500) + random(250, 1500);

    estado_button_anterior = estado_button; //Atualiza o estado do
botão afim de evitar o clique segurado

} //Fim RESET

void setup(){

  //Definindo as pinagens:
  pinMode(LEDG, OUTPUT);
  pinMode(LEDR, OUTPUT);
  pinMode(BUTTON, INPUT);

  //Definindo primeiros valores do game:
  tempo_topeira = random(400, 1500);
  tempo_topeira2 = random(400, 1500) + random(250, 1500);;

} //Fim SETUP

void loop(){

  //Lendo o tempo e atualizando e lendo entrada botão:
```

```
  tempo_atual = millis();
  estado_button = digitalRead(BUTTON);


  if(tempo_atual - tempo_anterior > tempo_topeira && estado_verde ==
0){ //If que acende o led verder

    estado_verde =! estado_verde;
    digitalWrite(LEDG, estado_verde);

  } //Fim if

  if(tempo_atual - tempo_anterior2 > tempo_topeira2 && estado_verde ==
1){ //If que verifica o tempo e se o usuário não clicou no botão,
acende vermelho + reset

    estado_verde =! estado_verde;
    digitalWrite(LEDG, estado_verde);

    for(int i = 0; i < 3; i++){ //Pisca de led

    digitalWrite(LEDR, HIGH);
    delay(200);
    digitalWrite(LEDR, LOW);
    delay(500);

    } //Fim for

    reset(); //Chama a função reset

  } //Fim if

  if(estado_button == 1 && estado_verde == 0 && estado_button_anterior
== 0){ //If que verifica se o botão foi pressionado antes do led verde
estar acesso

    digitalWrite(LEDR, HIGH);
    delay(2000);
    digitalWrite(LEDR, LOW);

    reset(); //Chama a função reset
```

```
  } //Fim if

  if(estado_button == 1 && estado_verde == 1 && estado_button_anterior
== 0){//If que verifica se o botão foi pressionado no tempo correto

  estado_verde =! estado_verde;
  digitalWrite(LEDG, estado_verde);

  for(int i = 0; i < 10; i++){ //Pisca de led

    digitalWrite(LEDR, HIGH);
    delay(50);
    digitalWrite(LEDR, LOW);
    delay(50);

    } //Fim for

  reset(); //Chama a função reset

  } //Fim if

  estado_button_anterior = estado_button; //Atualiza o estado do borão
afim de evitar o clique segurado

} //Fim LOOP
```