RELATÓRIO DE AVALIAÇÃO

Giuliano Bohn Benedeti
Keli Tauana Ruppenthal
Victor Dallagnol Bento

Objetivo: Escrever um programa usando a metodologia TDD e os drivers já desenvolvidos (LED, GPIO, Timer e PWM).

Descrição: O programa deve permitir ao usuário interagir com o LED através do botão com as seguintes funcionalidades: **1.** A cada toque no botão, o brilho do LED deve aumentar em 10% até o valor máximo (usando PWM), reiniciando em 10% na sequência. **2.** Ao pressionar o botão continuamente por mais de 2 segundos (contados com o Timer), o brilho é ajustado para o valor máximo. Se for pressionado novamente, volta à configuração de brilho anterior. **3.** Caso o usuário pressione o botão por 3 vezes seguidas em menos de 1 segundo, o LED entra em modo de pisca, com uma taxa de 5Hz. Neste modo, um novo pressionamento faz retorná-lo ao estado de configuração anterior.

Procedimento: A construção do projeto deu-se através da consulta dos exercícios anteriores, que foram adaptados conforme fosse necessário.

O primeiro trecho da avaliação utiliza-se de PWM. A lógica do nosso código dá-se da seguinte forma: inicializamos um contador em 1, então chamamos a função de configuração do PWM e da porta, isso dentro de um loop infinito utilizando o comando while. Seguindo foi criado um outro laço de while que funciona da seguinte forma: quando o botão for pressionado (true) ele verifica o contador. Se o contador já tiver atingido 10 (100%) , o pino 3 do pwm é setado com intensidade de 0xFFF (10%) e o contador é reinicializado para 1.

A luminosidade maxima fornecida pelo PWM corresponde por 0xFFFFF. Prosseguindo: caso o contador não tenha atingido o valor 10 então ele entra no else onde tem novamente uma condição que verifica se o botão está pressionado para

então ele setar o PWM de acordo com o contador, multiplicando o percentual 10% pelo número do contador. O contador então é incrementado.

Caso o botao nao esteja pressionado nada vai acontecer, por isso de não haver um else acompanhado do if. Segue abaixo um printscreen do código descrito no programa Notepad++

```
int main (void)
{
    system_init();

    int counter = 1;
    pwm_config(3, CONF_PWM_OUTPUT);
    porta_config(led, LED_0_PIN, PORT_PIN_DIR_OUTPUT);

    while (true){
        if (counter == 10){     // Periodo máximo do PWM = 0xFFFF
            pwm_set(3, 6553);   // Seta o pino e altera a razão cíclica para 10% do brilho 0xFFF
            counter = 1;        // Reseta contador
            delay_ms(1000);     // Delay de 1 segundo para coseguir ver LED
        }else{
            if(port_pin_get_input_level(BUTTON_0_PIN) == BUTTON_0_ACTIVE){  // Se botão estiver pressionado
                pwm_set(3, counter*6553);   // Seta o pino e altera a razão cíclica para +10% do brilho
                counter++;
            delay_ms(1000);     // Delay de 1 segundo para coseguir ver LED
            }
        }
    }

}
```

imagem 01 - algoritmo da primeira funcionalidade do exercício avaliativo

No segundo trecho foram criadas as seguintes variáveis: click (que recebe o numero de cliques pressionados pelo mouse), state (define qual o estado relativo aos cliques em que se encontra), conter (contador que vai definir a intensidade da luminosidade do LED), segundos (logicamente responsável pela contagem do tempo em segundos) e r_time (que retorna a função time).

Na nossa main são chamadas as funções de System_init (inicia o sistema), pwm_config (configuração do nosso PWM que está dentro de uma biblioteca), porta_config (configuração inicial do LED) e timer_config (que configura o nosso driver do timer para 2 segundos).

Próximo passo é realizar a contagem do timer através de uma função while infinita. Nesta mesma função está inserida uma condição muito importante: se o estado for o primeiro ele executa a primeira função. Caso o estado for o segundo ele

vai setar o brilho do LED como máximo pela função do PWM para então chamar a primeira função. Caso contrário no último else ele apenas chama a primeira função.

```
// Globais
int click = 0;      // Variável para pegar o numero de clicks
int state = 1;      // Define qual será a atividade
int counter = 1;    // Contador para aumentar o brilho do LED a cada 10%
int segundos = 0;   // Variável que controla quantos segundos passaram
uint32_t r_time;    // retorna da função time

int main(){
    system_init();                                      // Inicia sistema
    pwm_config(3, CONF_PWM_OUTPUT);                     // Configura PWM
    porta_config(led, LED_0_PIN, PORT_PIN_DIR_OUTPUT);  // Configura LED
    timer_config(1, 1000);                              // configura timer para 2 Segundos


    while(1){
        timer_start(1);     // Começa a contagem do timer

        if(state == 1)      // Começa no Estado 1
            ex_1();
        else if (state == 2)
            // Max brilho no LED
            pwm_set(3, 1);              // Numero máximo 0xFFFF (65535) divido por 1
            ex_1();
        else
            ex_1();         // Chama função 1 que verifica se deu click
    }
}
```

imagem 02 - imagem exibindo as variáveis globais e a função principal.

A primeira função (ex_1) funciona da seguinte maneira: se o contador chegou a 10 (100% do brilho) ele seta o pino (no nosso caso o pino 3) para os 10% (6553), reseta o contador e coloca aquele delay para permitir a visualização do LED.

Caso contrário, ou seja, o contador não estar com o valor final, a segunda condição vai setar o estado para 1, multiplicar o contador pelos 10% da luminosidade, incrementar o contador, incrementar o controle do clique, pegar o tempo com a função timer. O mesmo vai controlar o tempo até chegar em 1 segundo aproximadamente, e ao chegar, ele vai pular para a segunda função. E caso contrário (não esteja ainda em 1 segundo) irá então para a função de número 3 (LED piscando).

```
void ex_1(){
    if (counter == 10){         // Periodo máximo do PWM = 0xFFFF
            pwm_set(3, 6553);   // Seta o pino e altera a razão cíclica para 10% do brilho 0xFFF
            counter = 1;        // Reseta contador
            delay_ms(1000);     // Delay de 600 ms para coseguir ver LED
    }else{
        if(port_pin_get_input_level(BUTTON_0_PIN) == BUTTON_0_ACTIVE){  // Se botão estiver pressionado
            state = 1;
            pwm_set(3, counter*6553);   // Seta o pino e altera a razão cíclica para +10% do brilho
            counter++;                  // Incrementa contador
            click++;                    // Botão foi clicado
            r_time = timer_count(1)     // Pega tempo
            if (r_time == 1000){        // Verifica se deu 1s
                ex_2();                 // Salta para função ex_2
            }
        }else{
            state = 3;      // Recebe estado 3, mas o LED pode estar piscando ou no brilho maximo (state = 2)
                            // Para isso ele precisa sempre retornar a função ex_1 para verificar o botão e volt
        }
        delay_ms(600);      // Delay de 600 ms para coseguir ver LED
    }
}
```

imagem 03 - algoritmo continuando a função 1 do exercício.

Já a segunda função ela é a responsável para o caso de ocorrerem os 3 cliques em menos de um segundo: o LED piscar. Primeiramente é verificado o estado do botão e então cria-se uma condição: caso 3 cliques ocorram em menos de 3 segundos o estado é setado em 3, o timer é reiniciado e o led pisca ao chamar a função 3. Caso contrário aos 3 cliques então a variável que controla os segundos é incrementada e o timer iniciado e a variável de cliques reiniciada. Ao atingir os 2 segundos o estado do botão é averiguado e o mesmo se estiver setado como ligado faz a configuração do estado para 2.

E finalmente a terceira função que basicamente faz o seguinte, se o estado estiver setado em 3 (condição que ocorre dentro das funções anteriores) o LED é setado para piscas através da função de toggle.

```
void ex_2(){
    int btn_value;      // Recebe estado do botão

    if(click >= 3){     // Caso ocorra mais de 3 cliques por segundo
        state = 3;
        timer_start(1);     // Recomeça a contagem do segundo
        ex_3();             // pisca LED
    }else{
        segundos++;         // Incrementa segundo
        timer_start(1);     // Recomeça a contagem do segundo
        click = 0;          // Zera o numero de cliques
    }
    if (segundos == 2){     // Se atingiu 2 segundos
        btn_value = port_pin_get_input_level(LED0, &btn_value);     // Pega o estado do botão
        if(btn_value == ON) {       // Se ele estiver pressionado vai para o estado 2
            state = 2;
        }
    }
}

void ex_3(){
    if(state == 3){             // Se estiver no estado 3, pisca o LED
        led_toggle(led);
    }
}
```

imagem 04 - As funções 2 e 3 implementadas do algoritmo.