



**Universidade Federal do Sul e Sudeste do Pará
Faculdade de Computação e Engenharia Elétrica
Curso de Engenharia da Computação
Iago Costa das Flores**

**Microprocessadores e Microcontroladores
Experimento 9**

**Marabá
2021**

Microprocessadores e Microcontroladores
Experimento 9

Relatório apresentado no curso de Engenharia da Computação, turma de 2018 como obtenção de nota parcial na disciplina de microprocessadores e microcontroladores, ministrada pelo Professor Dr. Elton Alves.



Sumário

1 - Introdução	4
2 - Atividades	4
2.1 - Desenvolver um programa em linguagem C para controlar um cruzamento e ruas com 3 tempos, com o circuito correspondente no simulador Proteus e hardware na protoboard.	4
3 - Conclusão	9
4 - Referências	9

1 - Introdução

O Trabalho visa apresentar os códigos fontes e resultados de execução da atividade avaliativa a seguir:

1 - Desenvolver um código em C, utilizando o software Mplab, para: Inserir um botão, para acionar o semáforo; O semáforo será controlado pelo usuário; Após ser acionado, o sinal passará para a condição fechado (led vermelho) em seguida se iniciará contagem de tempo regressivo de 9 até 0; Em seguida o sinal será aberto.

2 - Atividades

As atividades demonstradas a seguir foram feitas com a ajuda do programa mplab e proteus para escrever e executar os códigos em assembly.

2.1 - Desenvolver um código em C, utilizando o software Mplab, para: Inserir um botão, para acionar o semáforo; O semáforo será controlado pelo usuário; Após ser acionado, o sinal passará para a condição fechado (led vermelho) em seguida se iniciará contagem de tempo regressivo de 9 até 0; Em seguida o sinal será aberto.

Código da atividade 01:

```
/*
 * File:   semaforo.c
 * Author: Iago
 *
 * Created on July 28, 2021, 10:50 AM
 */

#include <xc.h>
#pragma config FOSC = INTOSCIO    // Oscillator Selection bits (HS
oscillator: High-speed crystal/resonator on RA6/OSC2/CLKOUT and
RA7/OSC1/CLKIN)
#pragma config WDTE = OFF          // Watchdog Timer Enable bit (WDT
disabled)
#pragma config PWRTE = OFF         // Power-up Timer Enable bit (PWRT
disabled)
```



```
#pragma config MCLRE = OFF          // RA5/MCLR/VPP Pin Function Select
bit (RA5/MCLR/VPP pin function is MCLR)
#pragma config BOREN = OFF          // Brown-out Detect Enable bit (BOD
disabled)
#pragma config LVP = OFF           // Low-Voltage Programming Enable bit
(RB4/PGM pin has digital I/O function, HV on MCLR must be used for
programming)
#pragma config CPD = OFF           // Data EE Memory Code Protection bit
(Data memory code protection off)
#pragma config CP = OFF            // Flash Program Memory Code Protection
bit (Code protection off)

#define _XTAL_FREQ 4000000

#define LED1 RA0
#define LED2 RA1
#define LED3 RA2

#define LED4 RA3
#define LED5 RB4
#define LED6 RB0

#define LED7 RB1
#define LED8 RB2
#define LED9 RB3

void main (){

    PORTB=0x00;
    TRISB=0x00;
    PORTA=0x00;
    TRISA=0x00;

    while(1)
    {
        // semaforo 1 aberto e demais fechados
        LED1=0;
        LED2=0;
        LED3=1;

        LED4=1;
        LED5=0;
        LED6=0;
```

```
LED7=1;
LED8=0;
LED9=0;

__delay_ms(5000);

LED1=0;
LED2=1;
LED3=0;

LED4=1;
LED5=0;
LED6=0;

LED7=1;
LED8=0;
LED9=0;

__delay_ms(1000);

// semaforo 2 aberto e demais fechados
LED1=1;
LED2=0;
LED3=0;

LED4=0;
LED5=0;
LED6=1;

LED7=1;
LED8=0;
LED9=0;

__delay_ms(5000);

LED1=1;
LED2=0;
LED3=0;

LED4=0;
LED5=1;
LED6=0;
```

```
    LED7=1;
    LED8=0;
    LED9=0;

    __delay_ms(1000);

    // semaforo 3 aberto e demais fechados
    LED1=1;
    LED2=0;
    LED3=0;

    LED4=1;
    LED5=0;
    LED6=0;

    LED7=0;
    LED8=0;
    LED9=1;

    __delay_ms(5000);

    LED1=1;
    LED2=0;
    LED3=0;

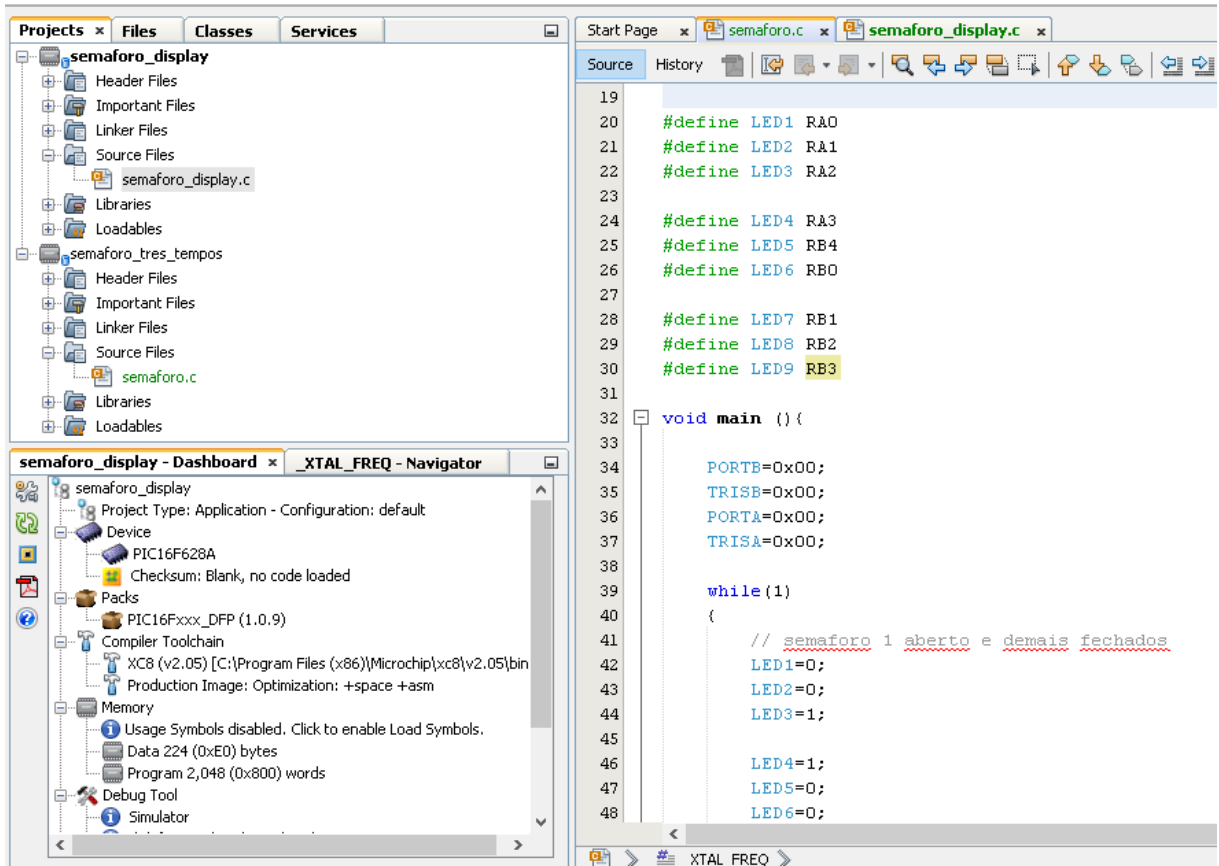
    LED4=1;
    LED5=0;
    LED6=0;

    LED7=0;
    LED8=1;
    LED9=0;

    __delay_ms(1000);

}
}
```

Na figura 01 é possível ver a declaração de variáveis e parte da função main do código fonte da atividade 01.



```

19
20 #define LED1 RA0
21 #define LED2 RA1
22 #define LED3 RA2
23
24 #define LED4 RA3
25 #define LED5 RB4
26 #define LED6 RB0
27
28 #define LED7 RB1
29 #define LED8 RB2
30 #define LED9 RB3
31
32 void main () {
33
34     PORTB=0x00;
35     TRISB=0x00;
36     PORTA=0x00;
37     TRISA=0x00;
38
39     while (1)
40     {
41         // semaforo 1 aberto e demais fechados
42         LED1=0;
43         LED2=0;
44         LED3=1;
45
46         LED4=1;
47         LED5=0;
48         LED6=0;

```

Figura 01: Código fonte da atividade 01

Na figura 02, temos a execução da atividade 01, com o esquema montado com o PIC16F628A.

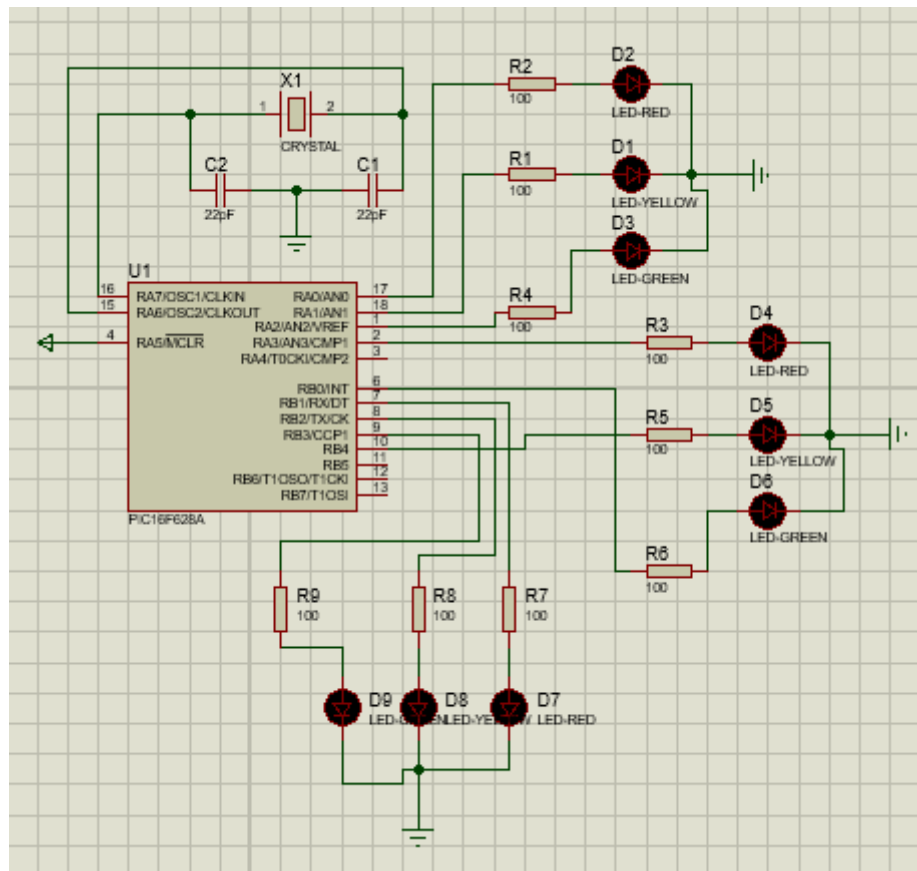


Figura 02: Execução da atividade 01

3 - Conclusão

Foram demonstradas as impressões do código e execução dos mesmos através das imagens apresentadas com suas devidas explicações. Os códigos fontes estão comentados e a atividade foi feita conforme o solicitado no comando do trabalho.

4 - Referências

José, D. S. **Desbravando o PIC - Ampliado e atualizado para PIC16F628A**. 7ª ed. Ed.Érica, 2003.