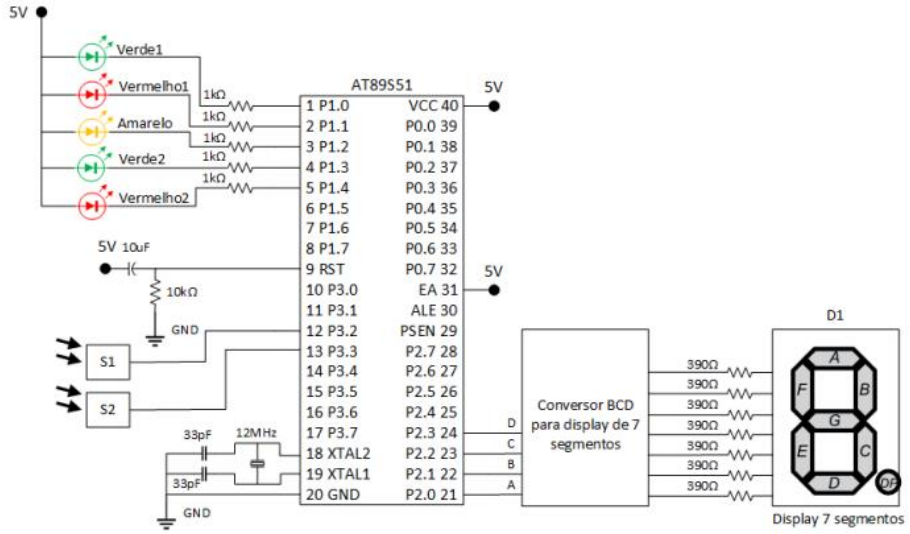
UNIVERSIDADE da MADEIRA

FCEE – Faculdade de Ciências Exatas e da Engenharia

Licenciatura em Engenharia Informática

Arquitetura de Computadores

3ºProjeto – Sistema de gestão de entradas em um parque de estacionamento



Docentes:

- Dionísio Barros
- Sofia Inácio
- Pedro Camacho
- Dino Vasconcelos

Discentes:

- Bjorn André Costa Foss – nº2048319
- Joana Andrade Azevedo – nº2076220

Conteúdo

1. Introdução

No âmbito da unidade curricular de Arquitetura de Computadores, foi-nos solicitado a criação de um sistema para gerir as entradas em um parque de estacionamento, de modo a verificarmos quantos lugares livres existem ao um automóvel entrar ou sair do parque.

Para um terceiro e último projeto desta unidade curricular, usamos o microcontrolador 8051, em específico, o AT89S51, com recurso ao software Keil Uvision5, onde pudemos simular os resultados pretendidos.

Para este projeto usamos dois tipos de linguagens, em C que é uma linguagem de alto nível, e Assembly que é uma linguagem de baixo.

2. Objetivos

O objetivo principal deste projeto é, como referido acima, o desenvolvimento de um programa que permite mostrar quantos lugares disponíveis é que existem na entrada e saída de um automóvel no estacionamento, através de um display, e também observar os comportamentos dos LEDs quando o carro passa nos sensores, de modo a verificar se o display está em sincronização com o estado dos LEDs.

3. Desenvolvimento

3.1. Cálculo do número de contagens/ciclos

Para realizar a contagem do tempo optou-se por utilizar o Timer no modo 1, ou seja, 16 bits sem autoreload.

Cálculo do número de ciclos/contagens:

65536us -10 000us = 55536 us (D8F0H).

10 000us*100 ciclos de máquina = 1 s ->variável contagens

Cálculo do TH0 e do TL0:

Como foi usado o modo1, não há autoreload então, quando há overflow, TH0 e TL0 tem que ser sempre inicializados com os valores: TH0 = 0xD8 e TL0 = 0xF0.

3.2. Desenvolvimento do código C

Para desenvolver o sistema de gerir as entradas em um parque estacionamento em linguagem C temos que incluir a biblioteca “reg51.h”, pois sem esta não era possível resolver o trabalho.

Utilizamos o #define para definir as contagens, o sbit para os leds em cada pino, o bit para os sensores e ainda foi definido o display, o número máximo de lugares no parque de estacionamento e duas variáveis globais que vão contar cada segundo que passa e contar os 5 segundos.

De seguida, é inicializado as interrupções na função Init (void). Começando pelo registo IE e ativamos o bit EA, que é responsável por ativar as interrupções como um todo. Em

seguida, ativamos os bits ET0, EX0 e EX1, que lidam com as interrupções externas e dos timers. Os valores do Timer Mode Configuration, calculados acima. O IP = 2 para ter maior prioridade no zero e no TCON é colocado os bits IT0 e IT1 a 1 de modo que as interrupções externas ficarem ativas em falling edge (transição descendente). Por fim colocamos no pino P2 o valor do display.

Na função main (void) é chamado a função Init (), para inicializar as interrupções e temos um loop infinito com os leds no seu estado inicial.

As duas funções Entrada (void) e Saída (void), tem a mesma estrutura os únicos pontos que mudam é a posição dos leds e a incrementação do display, retirar um valor ao display caso se um carro entra e aumentar caso um carro sai. Nestas funções temos um ciclo while que vai contar cada segundo que passa até chegar a 5 segundos fazendo led amarelo acender e desligar a cada vez que passa um segundo. No final de cada função o TR0 é colocado a zero, ou seja, para parar o timer.

Por último, temos as funções das interrupções. As interrupções externas 0 e 1, têm a mesma estrutura só que na externa 0 se o estacionamento tiver algum lugar livre entra, salta para função entrada e na externa 1 é o mesmo caso só que sai o carro através da função saída. A interrupção do tempo 0 volta a colocar os valores do TH0 e TL0, dado que este timer não volta a colocar o valor inicial, por não ser autoreload.

3.3. Desenvolvimento do código Assembly

No caso da programação em assembly para este sistema de gestão de entradas em parque de estacionamento, utilizou-se o conselho dos docentes e traduzimos o código em C para assembly, o que tornou mais fácil a sua realização.

Não há muito a referir neste ponto dado que se utilizou a mesma estrutura do código em C e esta já foi explicada no ponto 3.1.

4. Conclusão

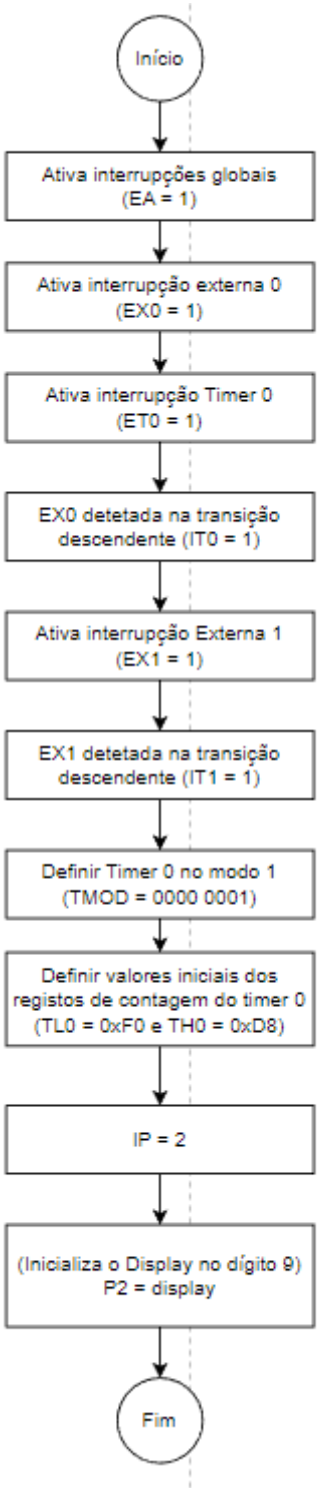
Após a conclusão do projeto, podemos afirmar que alcançamos todos os objetivos propostos, o que resultou em um programa de gestão de entradas em um parque de estacionamento desenvolvido em duas linguagens de programação diferentes.

A execução desse trabalho prático foi fundamental para consolidar os conhecimentos adquiridos tanto na teoria quanto na prática da disciplina. Além disso, permitiu ampliar nosso conhecimento em programação, especialmente nas linguagens assembly e C.

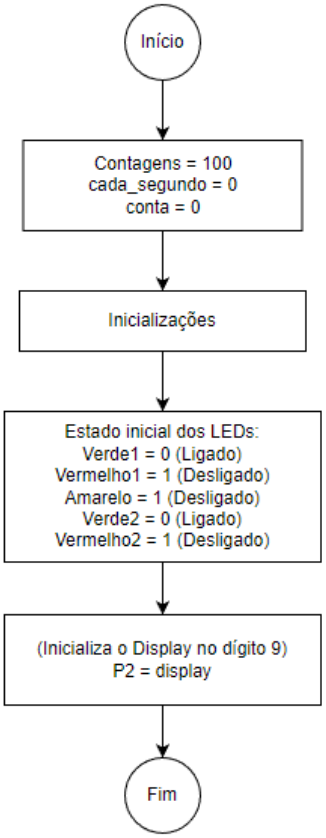
Durante a execução do projeto, adquirimos experiência e conhecimento, o que é uma parte natural desse tipo de trabalho. Enfrentamos erros e dificuldades ao longo do caminho, que se tornaram oportunidades para aprender e fortalecer nossos conhecimentos.

5. Anexo A – Fluxogramas

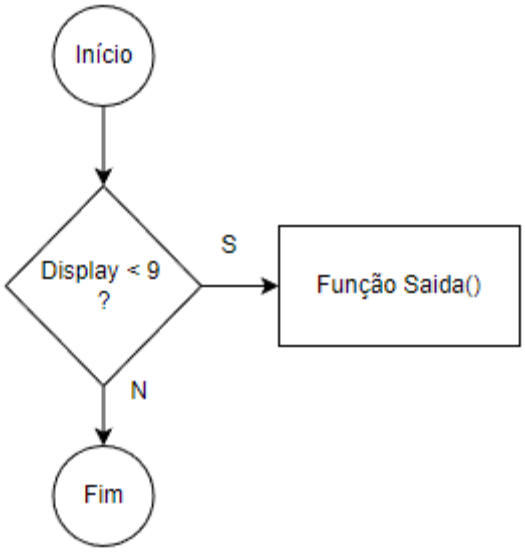
Inicializações:



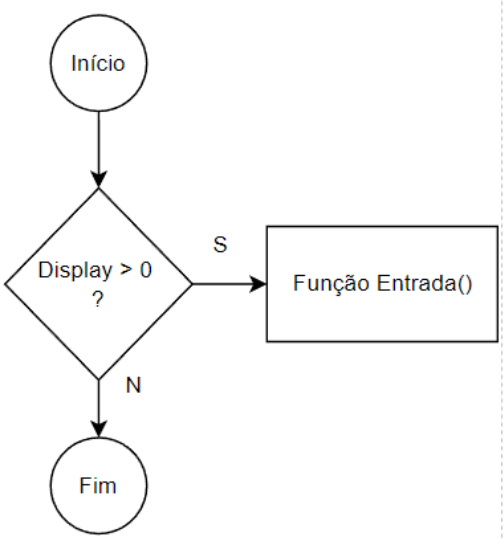
Programa Principal:



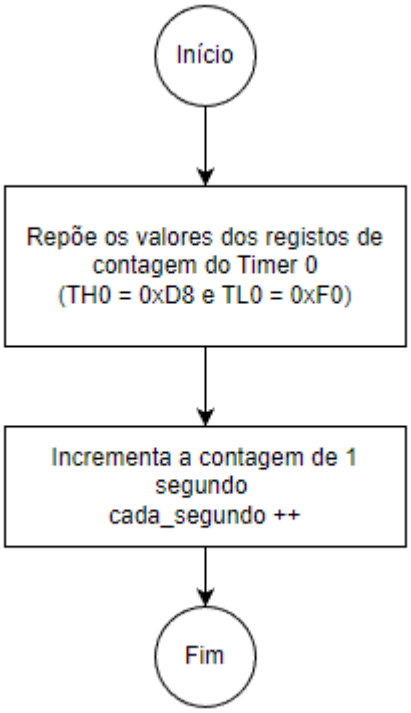
Interrupção Externa 1:



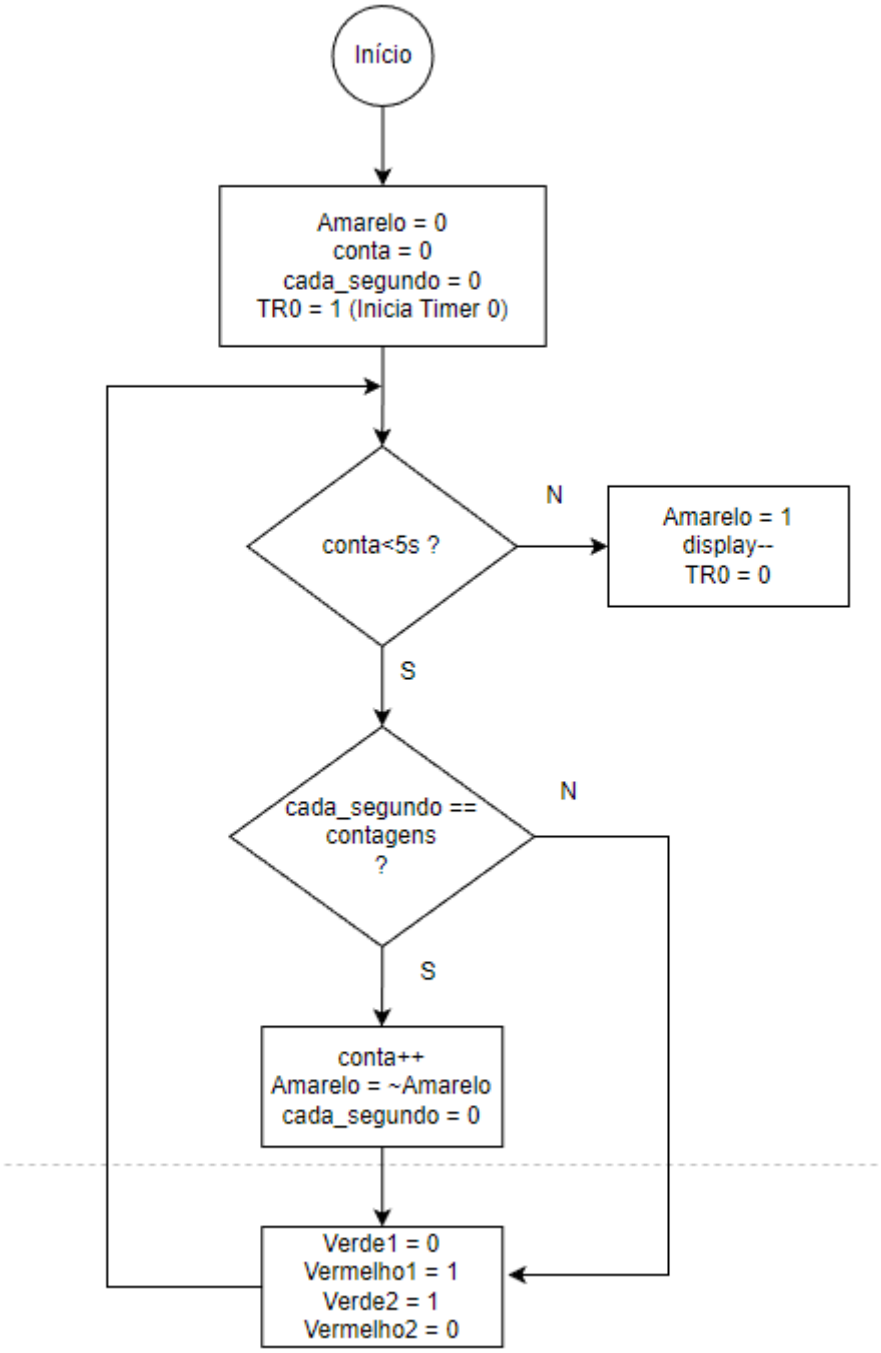
Interrupção Externa 0:



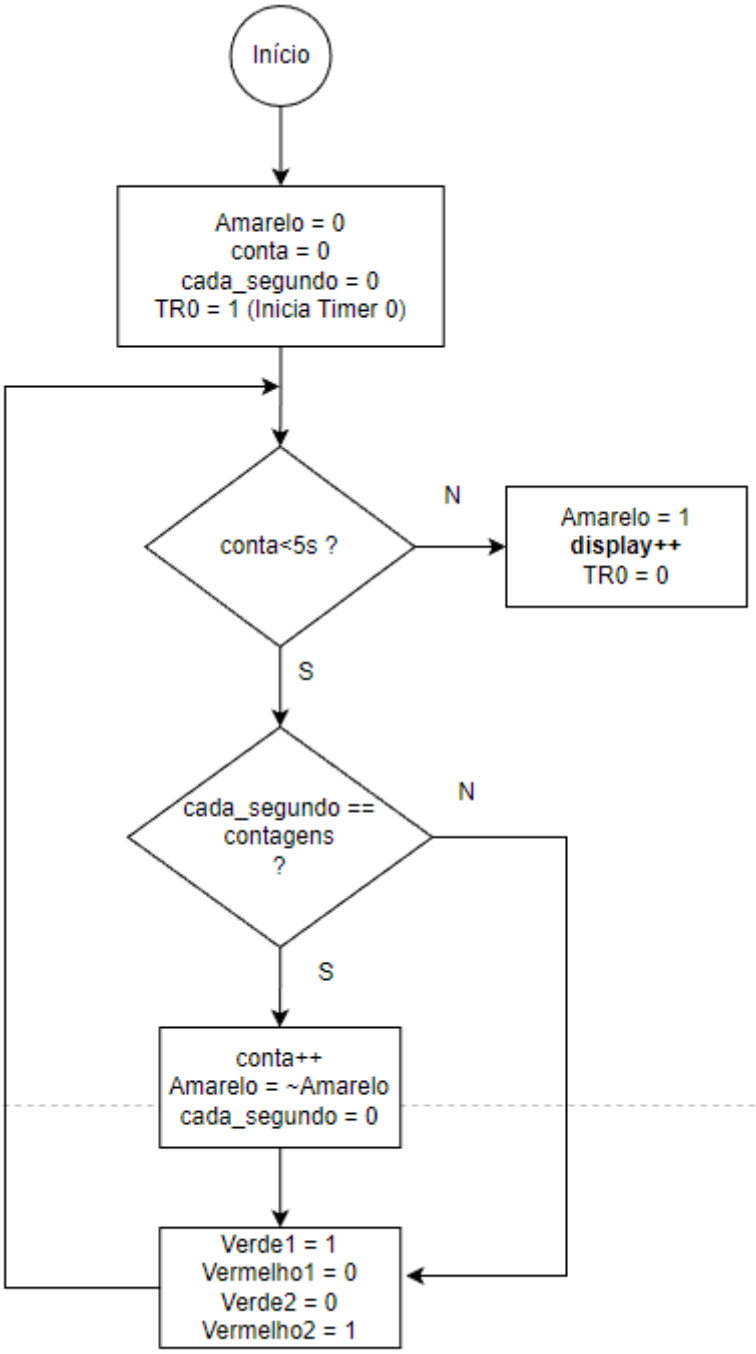
Interrupção Timer 0:



Entrada:

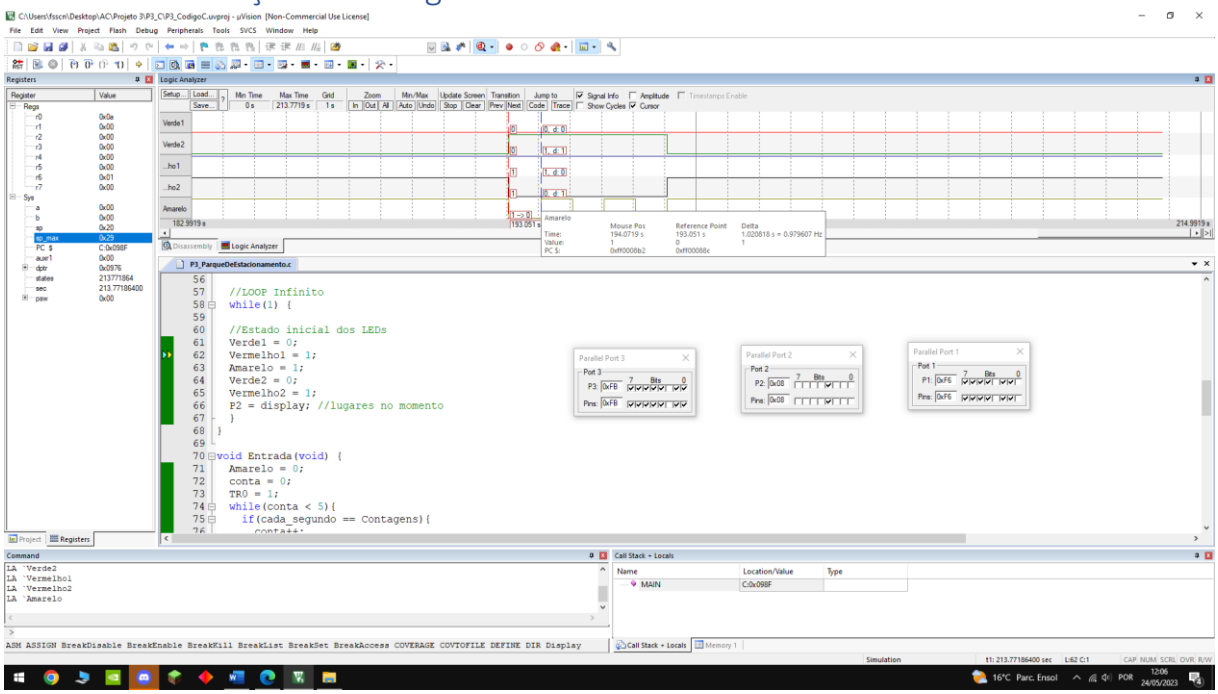

Início
Amarelo = 0
conta = 0
cada_segundo = 0
TR0 = 1 (Inicia Timer 0)
conta<5s ?
N
Amarelo = 1
display--
TR0 = 0
S
cada_segundo ==
contagens
?
N
S
conta++
Amarelo = ~Amarelo
cada_segundo = 0
Verde1 = 0
Vermelho1 = 1
Verde2 = 1
Vermelho2 = 0

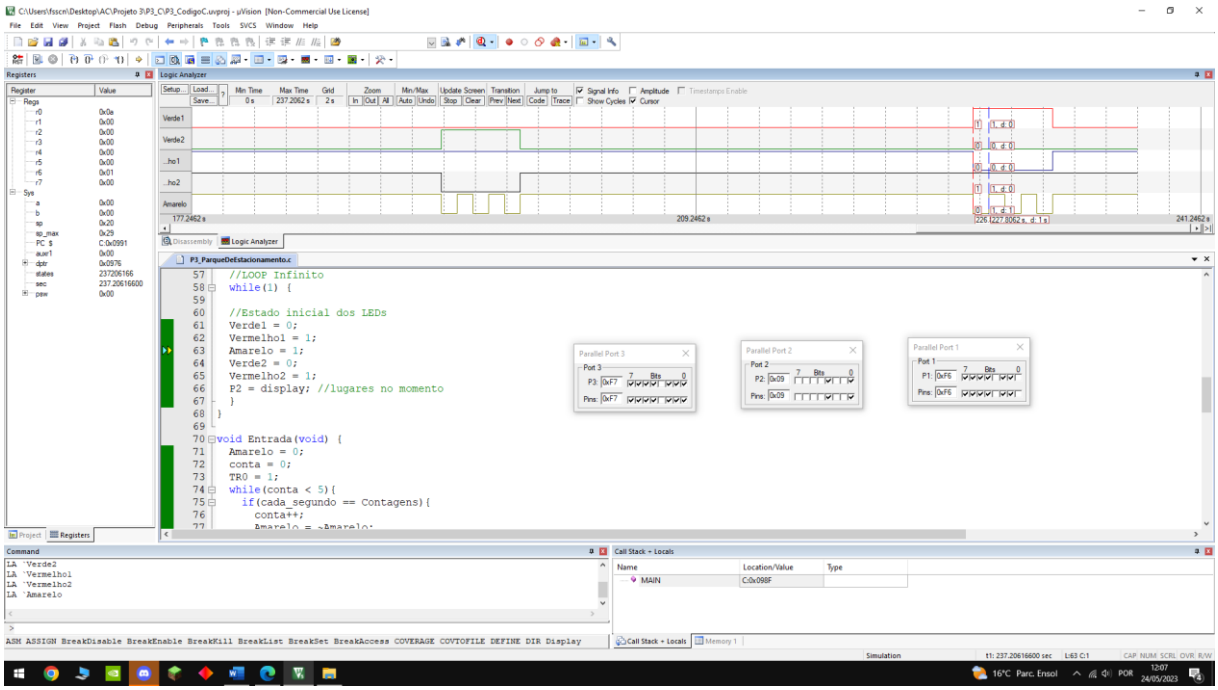
Saída:


Início
Amarelo = 0
conta = 0
cada_segundo = 0
TR0 = 1 (Inicia Timer 0)
conta<5s ?
N
Amarelo = 1
display++
TR0 = 0
S
cada_segundo == contagens ?
N
S
conta++
Amarelo = ~Amarelo
cada_segundo = 0
Verde1 = 1
Vermelho1 = 0
Verde2 = 0
Vermelho2 = 1

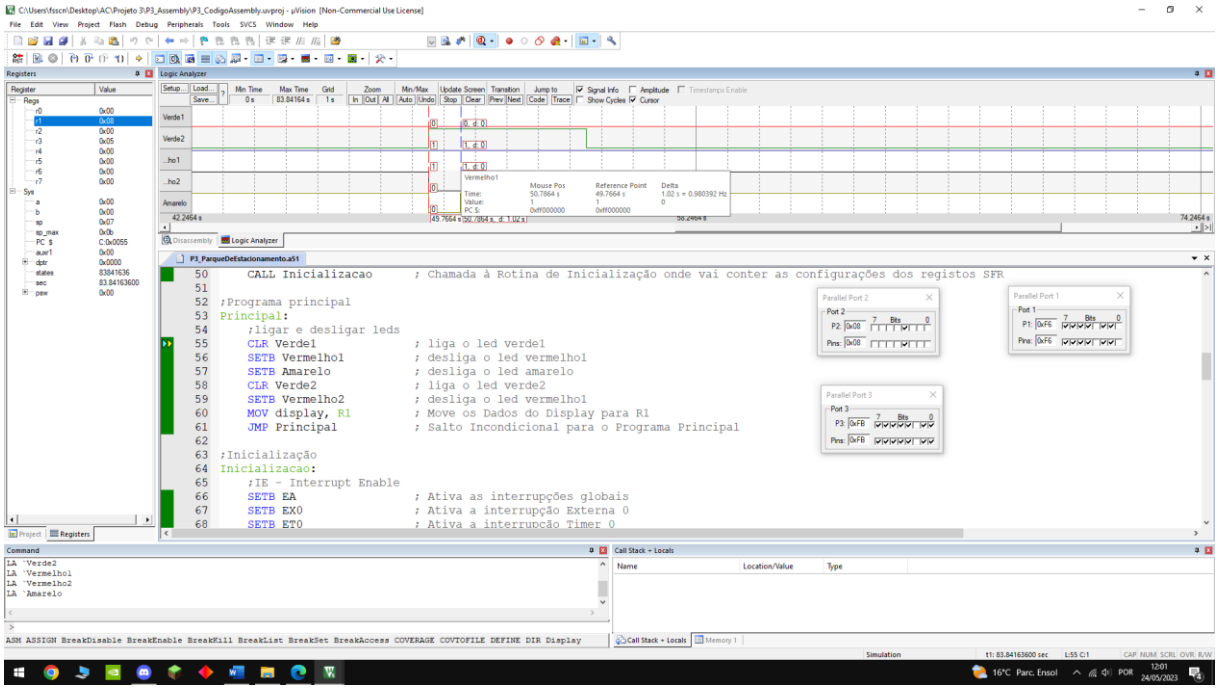
6. Anexo B – Simulação

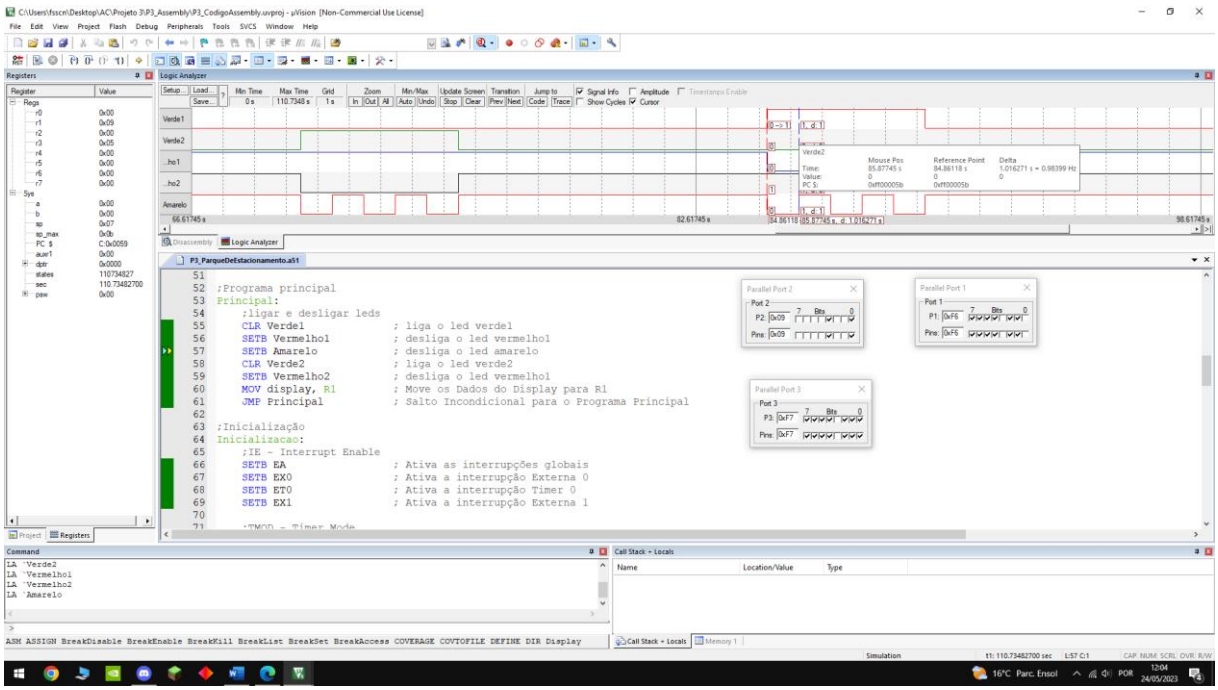
6.1. Simulação do Código em C





6.2. Simulação do Código em Assembly





7. Anexo C – Código

7.1. Código C

```
// P3 -> Arquitetura de Computadores

// Sistema de gest?o de entradas em um parque de estacionamento

// Linguagem C


#include <reg51.h>                                  // biblioteca do 8051
```

```
//ciclos
#define Contagens 100                    // 100 contagens * 10000us = 1s

//luzes
sbit Verde1 = P1^0;                      //LED Verde1
sbit Vermelho1 = P1^1;                   //LED Vermelho1
sbit Amarelo = P1^2;                     //LED Amarelo
sbit Verde2 = P1^3;                      //LED Verde2
sbit Vermelho2 = P1^4;                   //LED Vermelho2

//sensores
bit Sensor1 = 0;                         //Sensor1      -> P3^2
bit Sensor2 = 0;                         //Sensor2 -> P3^3

//display
int display = 9;                         //valores do display

//lugares de estacionamento
int lugar = 9;                           //N?mero m?ximo de
lugares dispon?veis

//Vari?veis Globais
int cada_segundo = 0;                    //conta cada segundo
int conta = 0;                           //contar os 5 segundos


void Init(void) {
        //IE
        EA = 1;
        //Ativa Interrup??es Globais
```

```
	EX0 = 1;
	//Ativa Interrup??o Externa 0

	ET0 = 1;                                                    //Ativa
Interrup??o Timer 0

	EX1 = 1;
	//Ativa Interrup??o Externa 1


	//TMOD - Timer Mode Configuration

	// 2^16 - 10000 = 55536 -> D8F0H

	TMOD |= 0x01;                                           //Timer Mode
1

	TH0 = 0xD8;                                                 //Bit
menos significativo no TH0 = 0xD8

	TL0 = 0xF0;                                                 //Bit
mais significativo em TL0 = 0xF0


	//IP

	IP = 2;
	//prioridade mais elevada no zero


	//TCON

	IT0 = 1;                                                    //Ser
detetado na Transi??o descendente

	IT1 = 1;


	P2 = display;                                           //Inicializa o
Display 9

}


//Programa Principal

void main(void) {                                       // Inicia o Programa
Principal


	//Inicializa??es

	Init();
```

```
	//LOOP Infinito
	while(1) {

	//Estado inicial dos LEDs
	Verde1 = 0;                                                    // Ativa
o LED Verde1
	Vermelho1 = 1;                                          // Desativa o LED
Vermelho1
	Amarelo = 1;                                                  // Desativa o
LED Amarelo
	Verde2 = 0;                                                    // Ativa
o LED Verde2
	Vermelho2 = 1;                                          // Desativa o LED
Vermelho2
	P2 = display;                                                 // lugares no
momento
	}
}

void Entrada(void) {
	Amarelo = 0;
	conta = 0;
	TR0 = 1;
	while(conta < 5){
                                        // ciclo while -> se a contagem for menor que 5
segundos
		if(cada_segundo == Contagens){
	// se a contagem de cada segundo for igual ao n?mero de contagens
			conta++;
                                                        // incrementa a contagem dos
5 segundos
			Amarelo = ~Amarelo;
                        // Altera o estado do LED Amarelo
			cada_segundo = 0;
```

```
                }
                //altera??o dos leds, caso entre um carro
                Verde1 = 0;
                                                        // Desliga o LED Verde1
                Vermelho1 = 1;
                                        // Liga o LED Vermelho1
                Verde2 = 1;
                                                        // Liga o LED Verde2
                Vermelho2 = 0;
                                        // Desliga o LED Vermelho2
        }
        Amarelo = 1;
        display--;
                                                        // Decrementa o n?mero do Display, ou
seja, ? ocupado mais um lugar no estacionamento
        TR0 = 0;
                                                                // Para o Timer 0
}

void Saida(void) {
        Amarelo = 0;
        conta = 0;
        TR0 = 1;
        while(conta < 5){
                                        // Se o contador n?o ultrapassar os 5 segundos
                if(cada_segundo == Contagens){
        // Cada segundo corresponde ao n?mero de contagens
                        conta++;
                                                                // Incrementa a contagem dos
5 segundos
                        Amarelo = ~Amarelo;
                                // Altera o estado do LED Amarelo
                        cada_segundo = 0;
                }
                //altera??o dos leds, caso entre um carro
```

```
                Verde1 = 1;
                                                        // Desliga o LED Verde1
                Vermelho1 = 0;
                                        // Desliga o LED Vermelho1
                Verde2 = 0;
                                                        // Desliga o LED Verde2
                Vermelho2 = 1;
                                        // Liga o LED Vermelho2
        }
        Amarelo = 1;
                                                // Ativa o LED Amarelo
        display++;
                                                        // Incrementa o valor do Display, ou
seja, existe mais um lugar livre no estacionamento
        TR0 = 0;
                                                                // Para o Timer 0
}



//Interrup??o Externa 0
void InterrupcaoExt0(void) interrupt 0 {                // De acordo com o Registo -> EX0 ? o
Bit 0
        if(display > 0) {
                                // Se o estacionamento tiver algum lugar livre
                Entrada();
                                                        // Chama a fun??o Entrada
        }
}


//Interrup??o Timer 0
void InterrupcaoTemp0(void) interrupt 1 {                       // De acordo com o Registo ->
ET1 ? o Bit 1
        TH0 = 0xD8;
                                                        // Byte mais Significativo: TH0 -> D8H
        TL0 = 0xF0;
                                                        // Byte menos Significativo: TL0 -> F0H
```

```
        cada_segundo++;
                                                    // Incrementa a contagem de 1 em 1 segundo
}


//Interrup??o Externa 1
void InterrupcaoExt1(void) interrupt 2 {                    // De acordo com o Registo -> EX1 ? o
Bit 2
        if(display < 9) {
                                        // Se o estacionamento tiver algum lugar ocupado
                Saida();
                                                            // Chama a fun??o Sa?da
        }
}
```

7.2. Código Assembly

```
; P3 -> Arquitetura de Computadores
; Assembly

;Definição de constantes

;luzes
Verde1          EQU  P1.0               ; LED Verde1
Vermelho1       EQU  P1.1               ; LED Vermelho1
Amarelo                 EQU  P1.2               ; LED Amarelo
Verde2          EQU  P1.3               ; LED Verde2
Vermelho2       EQU  P1.4               ; LED Vermelho2

;sensores
Sensor1                 EQU  P3.2               ; Sensor1 -> P3^2
Sensor2                 EQU  P3.3               ; Sensor2 -> P3^3
```

```
;ciclos
Contagens       EQU 100                 ; 10000us = 100contagens para chegar a 1s

;LUGARES DE ESTACIONAMENTO
lugar           EQU 9                   ; lotação máxima de 9 lugares no estacionamento

constante       EQU 5                   ; Constante para controlar as contagens de 1 segundo

;display
display         EQU P2                  ; display -> P2 (P2^0, P2^1, P2^2, P2^3)


; Depois do reset
CSEG AT 0000h                           ; 1º Endereço de Memória do Programa
JMP Inicio                              ; Salto Incondicional para o Início do Programa

; Se ocorrer a interrupção externa 0
CSEG AT 0003h
JMP  InterrupcaoExt0            ; Salto Incondicional para a Interrupção Externa 0

; Se ocorrer a interrupção externa 1
CSEG AT 00013h
JMP  InterrupcaoExt1            ; Salto Incondicional para a Interrupção Externa 1

; Tratamento da interrupção de temporização 0, para contar 10ms
CSEG AT 000Bh
JMP  InterrupcaoTemp0           ; Salto Incondicional para a Interrupção Timer 0


CSEG AT 0050h                           ; Programa inicia no endereço de Memória 50H
```

```
Inicio:
        MOV SP, #7                      ; Endereço da STACK POINTER
        CALL Inicializacao              ; Chamada à Rotina de Inicialização onde vai conter as
configurações dos registos SFR

;Programa principal
Principal:
        ;ligar e desligar leds
        CLR Verde1                      ; liga o led verde1
        SETB Vermelho1                  ; desliga o led vermelho1
        SETB Amarelo                    ; desliga o led amarelo
        CLR Verde2                      ; liga o led verde2
        SETB Vermelho2                  ; desliga o led vermelho1
        MOV display, R1                 ; Move os Dados do Display para R1
        JMP Principal                   ; Salto Incondicional para o Programa Principal

;Inicialização
Inicializacao:
        ;IE - Interrupt Enable
        SETB EA                                 ; Ativa as interrupções globais
        SETB EX0                        ; Ativa a interrupção Externa 0
        SETB ET0                        ; Ativa a interrupção Timer 0
        SETB EX1                        ; Ativa a interrupção Externa 1

        ;TMOD - Timer Mode
        MOV TMOD, #1                    ; Timer Mode 1 ->
        MOV TH0, #0xD8                  ; Bit menos significativo no TH0 = 0xD8
        MOV TL0, #0xF0                  ; Bit mais significativo em TL0 = 0xF0

        ;IP
        MOV IP, #2              ;prioridade mais elevada no zero
```

```
        ;TCON - Timer Control
        SETB IT0                        ; Ativa Flag da interrupção Timer 0
        SETB IT1                        ; Ativa Flag da interrupção Timer 0

        MOV R1, #9                      ; Inicializa o Display no Dígito 9

        RET                                     ; Retorno da Rotina Inicializacao


;Interrupção externa 0
InterrupcaoExt0:                ; Inicia a Rotina da Interrupção Externa 0
        CJNE R1, #0, Entrada    ; Compara se R1 != 0
        RETI                            ; Retorno da Interrupção

Entrada: ;controlar o ciclo dos 5 segundos
        CLR Amarelo                     ; Desliga o LED Amarelo
        MOV R3, #0                      ; Contagem dos segundos = 0
        MOV R4, #0                      ; Contagem de cada segundo = 0
        SETB TR0                        ; TR0 =1 -> Inicia o Timer 0
TempoAmarelo:
        CJNE R3, #5     , TrocaAmarelo ;Compara se R3 != 5
        SETB Amarelo            ; Ativa o LED Amarelo
        JMP FimEntrada                  ; Salto Incondicional para a etiqueta
FimEntrada

TrocaAmarelo:
        CLR Verde1                      ; Desliga o LED Verde1
        CLR Vermelho2           ; Desliga o LED Vermelho2
        SETB Vermelho1                  ; Liga o LED Vermelho1
        SETB Verde2                     ; Liga o LED Verde2
```

```
        CJNE R4, #Contagens, TrocaAmarelo
        CPL Amarelo                 ; nega o bit
        MOV R4, #0                  ; Inicia a Contagem de cada segundo
        INC R3                      ; Incrementa a contagem dos 5 segundos
        JMP TempoAmarelo            ; Salto Incondicional para a Etiqueta TempoAmarelo

FimEntrada:
        DEC R1                      ; LUGAR = LUGAR - 1
        CLR TR0                         ; Desativa Timer 0
        RETI                        ; Retorno da Interrupção

;Interrupção externa 1
InterrupcaoExt1:                    ; Inicia a Rotina da Interrupção Externa 1
        CJNE R1, #9, Saida          ; Compara se R1 != 9
        RETI                        ; Retorno da Interrupção

Saida: ;controlar o ciclo dos 5 segundos
        CLR Amarelo
        MOV R3, #0                  ; Contagem dos segundos = 0
        MOV R4, #0                  ; Contagem de cada segundo = 0
        SETB TR0                    ; TR0 = 1 -> Inicia o Timer 0
Tempo_Amarelo:
        CJNE R3, #5     , Troca_Amarelo     ;COMPARA
        SETB Amarelo                ; Liga o LED Amarelo
        JMP FimSaida                ; Salto Incondicional para a etiqueta FimSaida

Troca_Amarelo:
        CLR Verde2                  ; Desliga o LED Verde2
        CLR Vermelho1               ; Desliga o LED Vermelho1
        SETB Vermelho2              ; Liga o LED Vermelho2
        SETB Verde1                 ; Liga o LED Verde1
```

```
        CJNE R4, #Contagens, Troca_Amarelo   ; Compara se R4 != #Contagens
        CPL Amarelo                  ; nega o bit (complemento para 2)
        MOV R4, #0                   ; Inicia a Contagem de cada segundo
        INC R3                       ; Incrementa a contagem dos 5 segundos
        JMP Tempo_Amarelo            ; Salto Incondicional para a Etiqueta Tempo_Amarelo

FimSaida:
        INC R1                       ; LUGAR = LUGAR + 1
        CLR TR0                              ; Desativa Timer 0
        RETI                         ; Retorno da Interrupção

;Interrupção do timer
InterrupcaoTemp0:
        MOV TH0, #0xD8               ; Coloca o valor de TH0 -> MSB: D8H
        MOV TL0, #0xF0               ; Coloca o valor de TL0 -> LSB: F0H
        INC R4                       ; Incrementa a contagem de cada segundo
        RETI                         ; Retorno da Interrupção

END
```