UNIVERSIDADE da MADEIRA

2022/2023

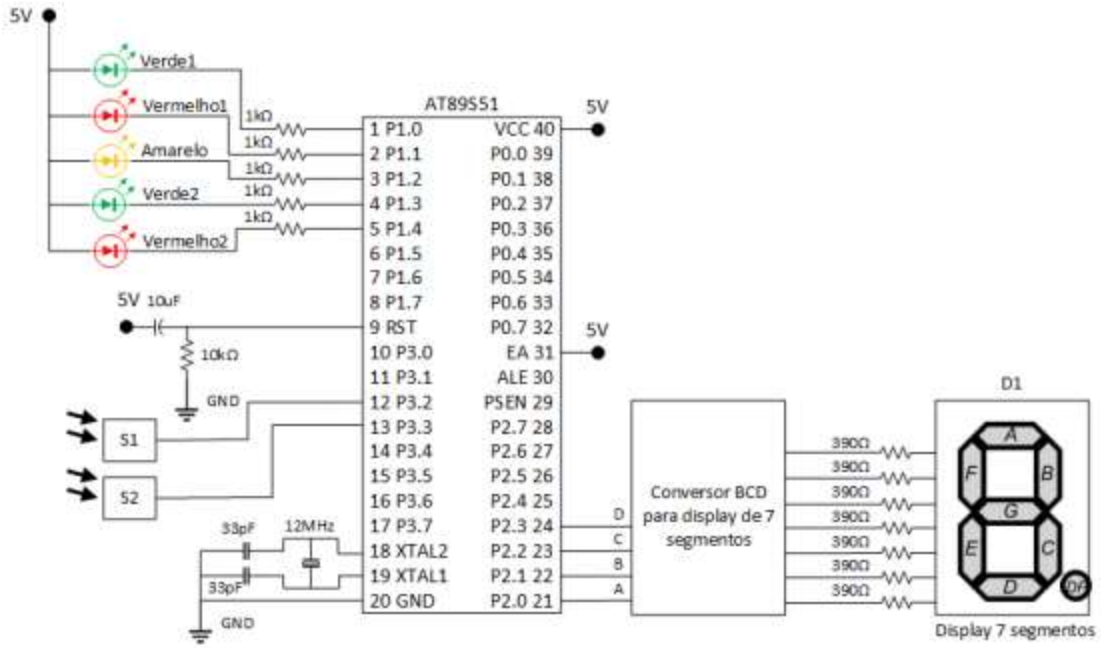
FCEE – LEI

ARQUITETURA DE COMPUTADORES

Díonisio Barros | Dino Vasconcelos | Pedro Camacho | Sofia Inácio

TRABALHO PRÁTICO 3

Sistema de gestão de entradas num parque de estacionamento



NUNO FERNANDES – 2082021 | JOÃO CASTRO - 2079521

ÍNDICE

OBJETIVOS

Este projeto, desenvolvido como parte da disciplina de Arquitetura de Computadores, apresenta um Sistema de Gestão de Entradas num Parque de Estacionamento, implementado utilizando a linguagem Assembly e a linguagem C no programa Keil uVision5. O objetivo central deste projeto é explorar os conceitos e princípios da arquitetura de computadores, aplicando-os na criação de um sistema eficiente e funcional.

O projeto procura abordar os desafios encontrados na gestão de entradas em parques de estacionamento, onde é necessário controlar o fluxo de veículos de forma precisa e segura. Através do uso de microcontroladores e sensores, o sistema desenvolvido é capaz de identificar e registar as entradas e saídas de veículos, garantindo um controlo adequado e eficiente.

A escolha das linguagens Assembly e C para o desenvolvimento deste sistema foi fundamentada na necessidade de explorar o conhecimento sobre a cadeira. A linguagem Assembly permite um controlo direto do hardware e a programação de instruções específicas para o microcontrolador, garantindo um alto desempenho e eficiência. Já a linguagem C oferece uma camada de abstração e facilita o desenvolvimento de funcionalidades mais complexas.

O Keil uVision5 foi selecionado como o ambiente de desenvolvimento para este projeto devido às suas poderosas ferramentas e recursos. Ele oferece suporte para a programação em Assembly e C, além de permitir a simulação e depuração do código, facilitando o processo de desenvolvimento e testes do sistema.

O projeto visa aprimorar a experiência dos utilizadores, oferecendo um controlo eficiente das entradas e saídas, reduzindo o tempo de espera e proporcionando uma gestão mais organizada. Além disso, a capacidade de gerar relatórios e estatísticas auxilia na tomada de decisões estratégicas para otimizar a utilização do parque de estacionamento.

DESCRIÇÃO DA SOLUÇÃO

LINGUAGEM C

Para desenvolver o programa fornecido, foi utilizado o Keil uVision5 e a linguagem C. O programa consiste em um sistema de gestão de entradas em um parque de estacionamento, utilizando botões e sensores para controlar o acesso e exibir a quantidade de vagas disponíveis.

No código fornecido, são definidos pinos para as diferentes luzes e para os segmentos de um display. Também são declaradas variáveis para controlar o número de vagas disponíveis no estacionamento e um contador.

A função "display" é responsável por exibir um número no display de sete segmentos. Ela utiliza uma matriz para mapear os segmentos correspondentes a cada dígito.

Na função principal "main", são configurados os registradores do timer e das interrupções. O display é inicializado com o número de vagas disponíveis.

Dentro do loop principal, são verificados os sensores de entrada e saída de veículos. Se o sensor1 for ativado e ainda houver vagas disponíveis, é iniciada a contagem do tempo. Durante os primeiros 5 segundos, as luzes são controladas de acordo com as condições especificadas. Após esse período, as luzes são alteradas e as variáveis são resetadas para receber um novo veículo. O número de vagas disponíveis é decrementado e exibido no display.

Da mesma forma, se o sensor2 for ativado e houver vagas ocupadas, é iniciada a contagem do tempo. Durante os primeiros 5 segundos, as luzes são controladas e, após esse período, as luzes são alteradas e as variáveis são resetadas. O número de vagas disponíveis é incrementado e exibido no display.

As interrupções externas 0 e 1 são tratadas pelas funções "interrupcaoExt0" e "interrupcaoExt1", respectivamente. A interrupção 0 é acionada quando o sensor1 é ativado e a interrupção 1 quando o sensor2 é ativado. A interrupção do timer 0 é tratada pela função "timer0" e é utilizada para contar o tempo.

Esse código implementa o sistema de gestão de entradas em um parque de estacionamento, controlando o acesso e exibindo a quantidade de vagas disponíveis. O sistema é responsivo às ações dos sensores e realiza as mudanças nas luzes de acordo com as condições especificadas.

LINGUAGEM ASSEMBLY

O programa começa com a definição de constantes e variáveis, onde são atribuídos rótulos aos pinos utilizados, como os LEDs e os segmentos de um display de sete segmentos. Também são definidas duas flags para sinalizar eventos e as variáveis para o número de lugares disponíveis e um contador.

Em seguida, o código principal é apresentado. A pilha é inicializada, são configurados os registradores de temporização (TMOD, TH0, TL0) para utilizar o Timer 0, e as interrupções são habilitadas. Além disso, os pinos são configurados e os registradores R0, R1, R2 e R3 são inicializados com valores específicos.

O programa então entra em um loop, verificando o estado dos sensores 1 e 2. Se a flag1 estiver ativada, o programa pula para a inicialização do Timer 0. Caso contrário, pula para o loop do Sensor 2. Isso significa que o programa está constantemente verificando os sensores para detectar a presença de veículos.

Quando um dos sensores é ativado, o programa inicializa o Timer correspondente (Timer 0 para o Sensor 1 e Timer 1 para o Sensor 2). Em seguida, verifica-se o valor do contador de segundos (R2) para determinar a lógica de sinalização.

Se o valor for igual a 5, o programa executa a função "cincoSegundos1" ou "cincoSegundos2", dependendo do Sensor. Essas funções são responsáveis por atualizar as luzes, decrementar ou incrementar o contador de lugares disponíveis e atualizar o display. Em seguida, o programa retorna ao loop do Sensor 1.

A função "clear" é chamada para limpar as sinalizações, reiniciar o contador de segundos, limpar as flags e reiniciar os registradores de temporização.

As funções "ativarAmarelo" e "desativarAmarelo" são responsáveis por ligar e desligar a luz amarela, respectivamente.

O programa também possui interrupções externas, sendo uma delas ativada quando o contador de lugares disponíveis (R3) é igual a 0, e a outra quando o contador é igual a 9. Quando essas interrupções ocorrem, as flags são ajustadas de acordo.

A interrupção de Timer 0 é ativada quando o temporizador dá overflow. Nesse caso, o programa realiza uma sequência de operações, como decrementar os registradores R0 e R1, limpar a flag de estouro do Timer 0 (TF0) e incrementar o contador de segundos (R2).

ANÁLISE DE RESULTADOS

Cada tópico dos requisitos do sistema de gestão de entradas no parque de estacionamento foi implementado com sucesso no projeto. Vamos detalhar como cada um foi abordado:

- Sensor S1 e Sensor S2:

Foram utilizados sensores óticos para detecção da passagem de veículos tanto na entrada quanto na saída do parque de estacionamento.

Os sensores funcionam corretamente, fornecendo um valor lógico '0' quando um veículo está bloqueando o feixe ótico e '1' quando não há veículo presente.

- Luzes Verde1 e Vermelho1:

As luzes Verde1 e Vermelho1 foram colocadas na entrada do parque de estacionamento.

A luz Verde1 encontra-se normalmente ligada, enquanto a luz Vermelho1 está desligada.

Quando um veículo entra no parque, a luz Verde1 é ligada e a luz Vermelho1 é desligada, sinalizando a entrada do veículo com sucesso.

- Luzes Verde2 e Vermelho2:

As luzes Verde2 e Vermelho2 foram colocadas na saída do parque de estacionamento.

A luz Verde2 encontra-se normalmente ligada, enquanto a luz Vermelho2 está desligada.

Quando um veículo sai do parque, a luz Verde2 é desligada e a luz Vermelho2 é ligada, indicando a saída do veículo com sucesso.

- Luz Amarela:

Foi implementada uma luz amarela adicional que pisca intermitentemente a cada segundo durante cinco segundos, quando um veículo entra ou sai do parque de estacionamento.

- Display de 7 segmentos (D1):

O display de 7 segmentos foi utilizado para mostrar o número de lugares disponíveis no parque de estacionamento. O parque possui uma lotação de 9 lugares.

A cada entrada de um veículo, o número de lugares disponíveis é decrementado. Da mesma forma, a cada saída, o número é incrementado.

O display é atualizado de acordo com as operações de entrada e saída de veículos, mostrando o número correto de lugares disponíveis entre 0 (parque cheio) e 9 (parque vazio) no início do programa.

Portanto, o sistema de gestão de entradas no parque de estacionamento foi implementado com sucesso, atendendo a todos os requisitos especificados. Os sensores, as luzes e o display de 7 segmentos funcionam corretamente, fornecendo uma gestão eficiente e precisa do estacionamento.

CONCLUSÃO

Concluindo, acreditamos que os objetivos do trabalho foram alcançados com sucesso, tanto na implementação em ambas as linguagens de programação quanto na visualização do correto funcionamento do programa utilizando a plataforma Keil uVision5.

A realização do trabalho não se mostrou muito complexa, uma vez que os alunos tinham um bom entendimento do funcionamento do processador e contaram com a elaboração prévia de fluxogramas, o que facilitou a compreensão do programa e seu funcionamento esperado. Os alunos ficaram satisfeitos com o programa desenvolvido, considerando-o simples e eficiente.

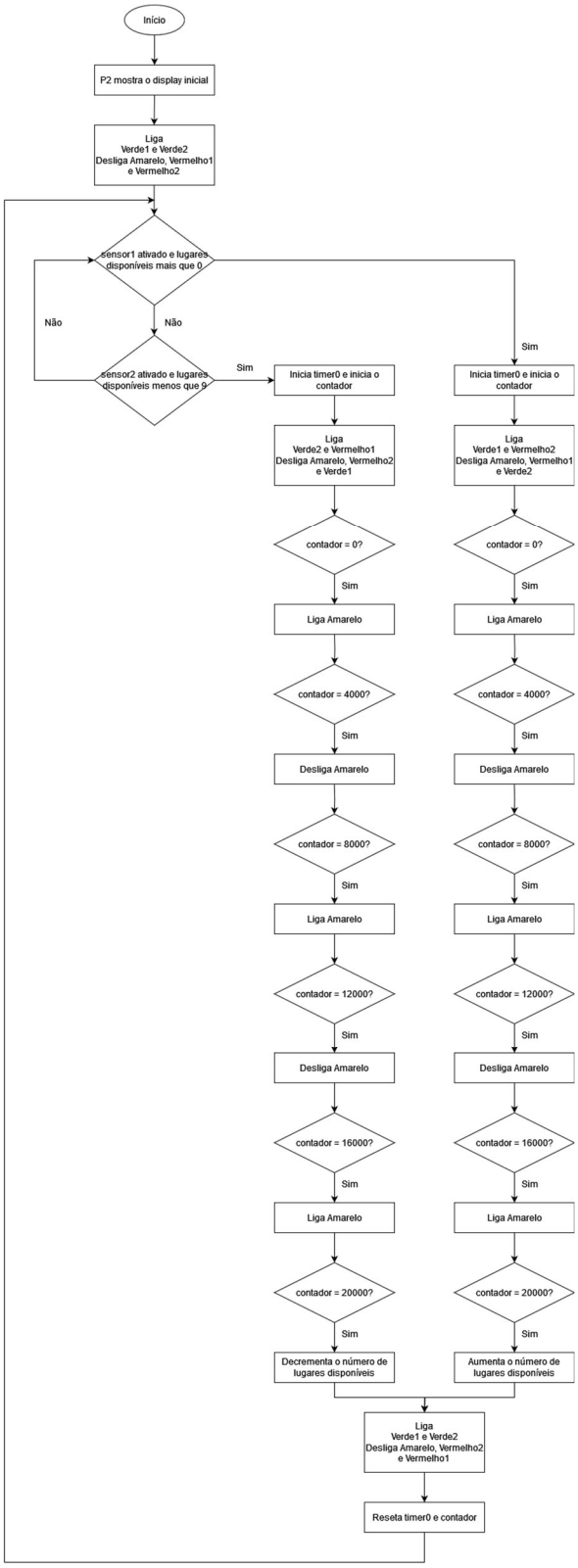
O desafio mais significativo foi o mapeamento do programa em C para a linguagem Assembly, pois exigia conhecimentos específicos sobre a conversão de instruções em C para instruções Assembly. No entanto, dada a simplicidade do programa em C, o processo de mapeamento foi facilitado pelos conhecimentos adquiridos sobre a conversão de instruções. Os alunos conseguiram realizar o mapeamento com sucesso, resultando em um programa funcional em Assembly.

BIBLIOGRAFIA

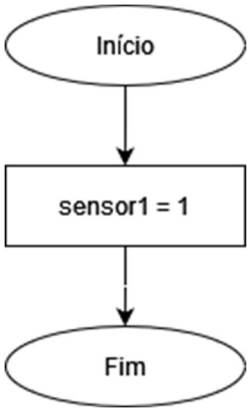
J. Delgado e C. Ribeiro, Arquitetura de Computadores, FCA - Editora de Informática, 2010.

ANEXO A

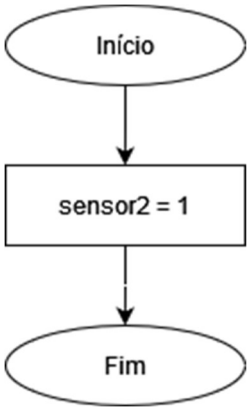
MAIN


Início
P2 mostra o display inicial
Liga
Verde1 e Verde2
Desliga Amarelo, Vermelho1
e Vermelho2
sensor1 ativado e lugares
disponíveis mais que 0
Não
Não
sensor2 ativado e lugares
disponíveis menos que 9
Sim
Sim
Inicia timer0 e inicia o
contador
Inicia timer0 e inicia o
contador
Liga
Verde2 e Vermelho1
Desliga Amarelo, Vermelho2
e Verde1
Liga
Verde1 e Vermelho2
Desliga Amarelo, Vermelho1
e Verde2
contador = 0?
contador = 0?
Sim
Sim
Liga Amarelo
Liga Amarelo
contador = 4000?
contador = 4000?
Sim
Sim
Desliga Amarelo
Desliga Amarelo
contador = 8000?
contador = 8000?
Sim
Sim
Liga Amarelo
Liga Amarelo
contador = 12000?
contador = 12000?
Sim
Sim
Desliga Amarelo
Desliga Amarelo
contador = 16000?
contador = 16000?
Sim
Sim
Liga Amarelo
Liga Amarelo
contador = 20000?
contador = 20000?
Sim
Sim
Decrementa o número de
lugares disponíveis
Aumenta o número de
lugares disponíveis
Liga
Verde1 e Verde2
Desliga Amarelo, Vermelho2
e Vermelho1
Reseta timer0 e contador

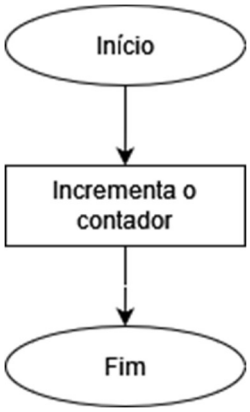
INTERRUPÇÃO EXT. 0



INTERRUPÇÃO EXT. 1



TIMER 0



ANEXO B

LINGUAGEM C

```
#include <reg51.h>

//-----------------------------------Constantes e Variáveis----------------------
-------------

sbit Verde1 = P1 ^ 0;
sbit Vermelho1 = P1 ^ 1;
sbit Amarelo = P1 ^ 2;
sbit Verde2 = P1 ^ 3;
sbit Vermelho2 = P1 ^ 4;
sbit segA = P2 ^ 0;
sbit segB = P2 ^ 1;
sbit segC = P2 ^ 2;
sbit segD = P2 ^ 3;

bit sensor1 = 0;
bit sensor2 = 0;

unsigned int availablePlaces = 9;
unsigned int counter = 0;

//-----------------------------------Funções-------------------------------------

void display(unsigned int num)
{
    unsigned segments[10][4] = {
        {0, 0, 0, 0}, // 0
        {0, 0, 0, 1}, // 1
        {0, 0, 1, 0}, // 2
        {0, 0, 1, 1}, // 3
        {0, 1, 0, 0}, // 4
        {0, 1, 0, 1}, // 5
        {0, 1, 1, 0}, // 6
        {0, 1, 1, 1}, // 7
        {1, 0, 0, 0}, // 8
        {1, 0, 0, 1}, // 9
    };

    segA = segments[num][3];
    segB = segments[num][2];
    segC = segments[num][1];
    segD = segments[num][0];
```

```
}

void main(void)
{
    TMOD = 0x02;
    TH0 = 0x06;
    TL0 = 0x06;
    ET0 = 1;
    EA = 1;
    EX0 = 1;
    IT0 = 1;
    EX1 = 1;
    IT1 = 1;

    display(availablePlaces);

    Verde1 = 0;    // Verde1 está ligado
    Vermelho1 = 1; // Vermelho1 está desligado
    Amarelo = 1;   // Amarelo está desligado
    Verde2 = 0;    // Verde2 está ligado
    Vermelho2 = 1; // Vermelho2 está desligado

    while (1)
    {
        if (sensor1 && availablePlaces != 0)
        { // Está a entrar um carro e tem lugares disponíveis
            TR0 = 1;

            // Durante 5 segundos:
            if (counter < 20000)
            {
                Verde1 = 0;    // Liga o Verde1
                Vermelho1 = 1; // Desliga o Vermelho1
                Verde2 = 1;    // Desliga o Verde2
                Vermelho2 = 0; // Liga o Vermelho2

                                if (counter == 0)
                                {
                                    Amarelo = 0;
                                } else {
                                if (counter % 4000 == 0)
                                {
                                        Amarelo = !Amarelo;
                                }
                                }
```

```
        }

        // Após os 5 segundos:
        if (counter == 20000)
        {
            Verde1 = 0;    // Liga o Verde1
            Vermelho1 = 1; // Desliga o Vermelho1
            Amarelo = 1;   // Desliga o Amarelo
            Verde2 = 0;    // Liga o Verde2
            Vermelho2 = 1; // Desliga o Vermelho2

            counter = 0;
            TR0 = 0;
            TH0 = 0x06;
            TL0 = 0x06;
            sensor1 = 0;
            sensor2 = 0;

            availablePlaces--;
            display(availablePlaces);
        }
    }

    if (sensor2 && availablePlaces != 9)
    { // Está a sair um carro
        TR0 = 1;

        // Durante 5 segundos:
        if (counter < 20000)
        {
            Verde1 = 1;    // Desliga o Verde1
            Vermelho1 = 0; // Liga o Vermelho1
            Verde2 = 0;    // Liga o Verde2
            Vermelho2 = 1; // Desliga o Vermelho2

            if (counter == 0)
            {
                Amarelo = 0; // Liga o Amarelo
            }
            if (counter == 4000)
            {
                Amarelo = 1; // Desliga o Amarelo
            }
```

```
                if (counter == 8000)
                {
                    Amarelo = 0; // Liga o Amarelo
                }
                if (counter == 12000)
                {
                    Amarelo = 1; // Desliga o Amarelo
                }
                if (counter == 16000)
                {
                    Amarelo = 0; // Liga o Amarelo
                }
            }

            // Após os 5 segundos:
            if (counter == 20000)
            {
                Verde1 = 0;    // Liga o Verde1
                Vermelho1 = 1; // Desliga o Vermelho1
                Amarelo = 1;   // Desliga o Amarelo
                Verde2 = 0;    // Liga o Verde2
                Vermelho2 = 1; // Desliga o Vermelho2

                counter = 0;
                TR0 = 0;
                TH0 = 0x06;
                TL0 = 0x06;
                sensor2 = 0;
                sensor1 = 0;

                availablePlaces++;
                display(availablePlaces);
            }
        }
    }
}

void interrupcaoExt0(void) interrupt 0
{
    sensor1 = 1;
}

void interrupcaoExt1(void) interrupt 2
{
    sensor2 = 1;
```

```
}

void timer0(void) interrupt 1
{
    counter++;
}
```

LINGUAGEM ASSEMBLY

;----------Constantes e Variáveis----------

Verde1 EQU P1.0

Vermelho1 EQU P1.1

Amarelo EQU P1.2

Verde2 EQU P1.3

Vermelho2 EQU P1.4

segA EQU P2.0

segB EQU P2.1

segC EQU P2.2

segD EQU P2.3

flag1 BIT 0

flag2 BIT 1

availablePlaces EQU 9

counter EQU 0

;----------Placements----------

CSEG AT 0000H

```
        JMP main

CSEG AT 0003H
        JMP interrupcaoExt0

CSEG AT 000BH
        JMP timer0

CSEG AT 0013H
        JMP interrupcaoExt1

;----------------------------------Código Principal---------------------------------

CSEG AT 0050H
main:
        MOV SP, #8
   MOV TMOD, #0x02                          ; Inicializacao do timer e das suas
interrupcoes
   MOV TH0, #0x06                           ; *
   MOV TL0, #0x06                           ; *
        MOV IP, #00000010b                  ; *
   SETB ET0                                 ; *
   SETB EA                                  ; *
   SETB EX0                                 ; *
   SETB IT0                                 ; *
   SETB EX1                                 ; *
   SETB IT1                                 ;
*********************************************
```

```
        MOV R0, #20
        MOV R1, #200
        MOV R2, #counter
    MOV R3, #availablePlaces

    CLR Verde1                          ; Verde1 esta ligado
    SETB Vermelho1                      ; Vermelho1 esta desligado
    SETB Amarelo                        ; Amarelo esta desligado
    CLR Verde2                          ; Verde2 esta ligado
    SETB Vermelho2                      ; Vermelho2 esta desligado
        JMP loopSensor1

loopSensor1:
        JB flag1, inicializaTimer0      ; Verifica se a flag1 esta ativada
        JMP loopSensor2

loopSensor2:
        JB flag2, inicializaTimer1      ; Verifica se a flag2 esta ativada
        JMP loopSensor1

inicializaTimer0:                       ; Esta a entrar um carro e tem lugares
disponiveis
        SETB TR0
        JMP verificaSegundos

inicializaTimer1:
        SETB TR0
        JMP verificaSegundos2
```

```
verificaSegundos:
        CJNE R2, #5, menorCincoSegundos1 ; Verifica se e menor que 5 segundos
        JMP cincoSegundos1

verificaSegundos2:
        CJNE R2, #5, menorCincoSegundos2 ; Verifica se e menor que 5 segundos
        JMP cincoSegundos2

menorCincoSegundos2:                    ; Durante 5 segundos
        SETB Verde1                         ; Liga o Verde1
        CLR Vermelho1                       ; Desliga o Vermelho1
        CLR Verde2                          ; Desliga o Verde2
        SETB Vermelho2                      ; Liga o Vermelho2
        JMP seForZeroSegundo2

menorCincoSegundos1:
        CLR Verde1                          ; Verde1 esta ligado
        SETB Vermelho1                      ; Vermelho1 esta desligado
        SETB Verde2                         ; Verde2 esta desligado
        CLR Vermelho2                       ; Liga o Vermelho2

; Ativacao ou desativacao do amarelo dependendo dos segundos
seForZeroSegundo:
        CJNE R2, #0, seForUmSegundo
        CALL ativarAmarelo
        JMP verificaSegundos
seForUmSegundo:
        CJNE R2, #1, seForDoisSegundos
```

```
        CALL desativarAmarelo
        JMP verificaSegundos
seForDoisSegundos:
        CJNE R2, #2, seForTresSegundos
        CALL ativarAmarelo
        JMP verificaSegundos
seForTresSegundos:
        CJNE R2, #3, seForQuatroSegundos
        CALL desativarAmarelo
        JMP verificaSegundos
seForQuatroSegundos:
        CALL ativarAmarelo
        JMP verificaSegundos

; Ativacao ou desativacao do amarelo dependendo dos segundos
seForZeroSegundo2:
        CJNE R2, #0, seForUmSegundo2
        CALL ativarAmarelo
        JMP verificaSegundos2
seForUmSegundo2:
        CJNE R2, #1, seForDoisSegundos2
        CALL desativarAmarelo
        JMP verificaSegundos2
seForDoisSegundos2:
        CJNE R2, #2, seForTresSegundos2
        CALL ativarAmarelo
        JMP verificaSegundos2
seForTresSegundos2:
```

```
        CJNE R2, #3, seForQuatroSegundos2
        CALL desativarAmarelo
        JMP verificaSegundos2
seForQuatroSegundos2:
        CALL ativarAmarelo
        JMP verificaSegundos2

cincoSegundos1:
        CALL clear
        DEC R3                          ; Decrementa o numero de
lugares disponiveis
        MOV P2, R3                      ; Mostrar no Display
        JMP loopSensor1                 ; Volta ao loop principal

cincoSegundos2:
        CALL clear
        INC R3                          ; Incrementa o numero de
lugares disponiveis
        MOV P2, R3                      ; Mostrar no Display
        JMP loopSensor1                 ; Volta ao loop principal

clear:
        CLR Verde1                      ; Verde1 esta ligado
    SETB Vermelho1                  ; Vermelho1 esta desligado
    SETB Amarelo                    ; Amarelo esta desligado
    CLR Verde2                      ; Verde2 esta ligado
    SETB Vermelho2                  ; Vermelho2 esta desligado
        MOV R2, #0
```

```
        CLR TR0                                         ; Reiniciar o Timer e as flags
de input
        CLR flag1                                       ; *
        CLR flag2                                       ; *
   MOV TH0, #0x06                                ; *
   MOV TL0, #0x06                                ;
**********************************
        RET

desativarAmarelo:
        SETB Amarelo
        RET

ativarAmarelo:
        CLR Amarelo
        RET

interrupcaoExt0:
   CJNE R3, #0, ativaflag1                      ; Verifica se o numero de lugares disponiveis e 0
para poder a seguir colocar um carro ou nao
        CLR flag1
   RETI

ativaflag1:
        SETB flag1
        RETI

interrupcaoExt1:
    CJNE R3, #9, ativaflag2                 ; Verifica se o numero de lugares disponiveis e 9 para poder
a seguir retirar um carro ou nao
```

```
        CLR flag2
    RETI

ativaflag2:
        SETB flag2
        RETI

timer0:
        DJNZ R0, final
        MOV R0, #20
        DJNZ R1, final
        CLR TF0
        INC R2                                  ; Incrementa o valor dos segundos apos passar um ciclo de 4000 ( 20 * 200 )
        MOV R1, #200

final:
    RETI

END
```