

MICROCONTROLADORES
FLUXOGRAMA E PROGRAMAÇÃO EM ASSEMBLY APLICADOS A FAMÍLIA
MICROCONTROLADORES MCS-51 DA INTEL

Relatório apresentado ao Curso de Microprocessadores e Microcontroladores, Setor de Engenharia Elétrica, Centro Universitária Inaciana Padre Sabóia de Medeiros.

Professor Orientador André Luiz Perin

IGOR BARROS VIANA

11.117.896-8

LUCAS PASTORI

11.117.775-4

CARLOS HENRIQUE

11.117.881-0

TURMA 725

SÃO BERNARDO DO CAMPO

2020

Sumário

1 OBJETIVO	3
2 METODOLOGIA.....	3
3 DESENVOLVIMENTO.....	4
4 SIMULAÇÃO	10
5 CONCLUSÃO E CONSIDERAÇÕES	23

OBJETIVO

O objetivo desse experimento foi desenvolver um projeto de aplicação real utilizando as ferramentas do microcontrolador 8051 e a placa de desenvolvimento. Foram usadas chaves, LEDs e Timers para criar toda a interface de utilização e visualização.

O projeto então teve como objetivo desenvolver um sistema de semáforo para atender um cruzamento, ou seja, ele consiste em administrar o comportamento dos carros e pedestres para que tudo ocorra bem e não aconteça acidentes.

METODOLOGIA

Para começar o desenvolvimento do projeto, algumas coisas foram definidas antes de dar seguimento à lógica. O semáforo deve ter 6 luzes, duas de cada cor (Verde, Amarelo, Vermelho), sendo uma de cada cor para cada via. Também há dois sensores de movimento, um para cada via, e o botão do pedestre.

Foi definido então que as luzes seriam representadas pelos LEDs do PORT1 [bit5 vermelho leste, bit4 amarelo leste, bit3 verde leste, bit2 vermelho norte, bit1 amarelo norte, bit0 verde norte], assim como os atuadores foram definidos como chaves do PORT0 [bit2 Pedestre passando, bit1 Sensor leste, bit0 Sensor norte]

Para o Timer que controla toda a temporização do semáforo foi definido o T0 como modo 0, contador de 16 bits. O tempo padrão de luz verde é de 5 segundos, de amarelo é de 1 segundo e o tempo de vermelho é a soma dos outros dois, pois caso haja movimento de carros detectado pelo sensor na respectiva via, o tempo de verde dessa via é aumentado para 8 segundos, e caso o pedestre aperte o botão os dois sinais fecham e o pedestre tem 5 segundos para atravessar.

Todas as detecções de chaves são feitas nas transições entre estados padrões, ou seja, os atuadores só influenciam quando o sistema se encontra em um dos estados de amarelo aceso, assim como no estado de pedestre.

Abaixo estão representadas todas as contas e resultados dos valores utilizados.

Cálculo de tempos para T0 em modo 0 (16bits de contagem) :

Frequência do cristal: 12MHz

$F/12 \rightarrow T = 1 \text{ us}$

Tempo máx = 65536 us

Desejado = 50000 us = 50 ms

Diferença = 15536 us

Valor em hexa = #3CB0h \rightarrow TH0 = #3Ch ; TL0 = #0B0h

Tempo de amarelo = 1s = 20 (#14h) interrupções de 50 ms

Tempo de verde = 5s = 100 (#64h) interrupções de 50 ms

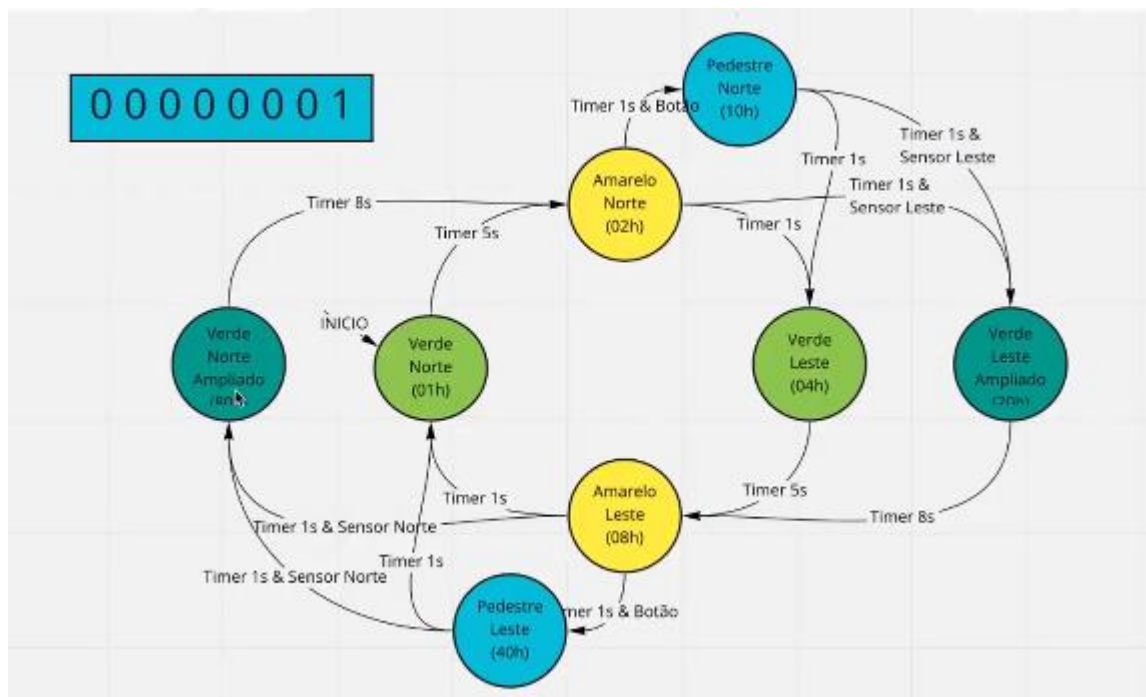
Tempo de vermelho = Tempo de verde + Tempo de amarelo

Tempo de pedestre = 5s = 100 (#64h) interrupções de 50 ms

Tempo de verde aumentado = 8s = 160 (#A0h) interrupções de 50 ms

DESENVOLVIMENTO

No desenvolvimento da lógica, um diagrama de estados básico deu o início para todo o resto, onde chegamos com o professor num sistema bem coerente e funcional. Abaixo está representada a imagem do diagrama de estados desenvolvido.

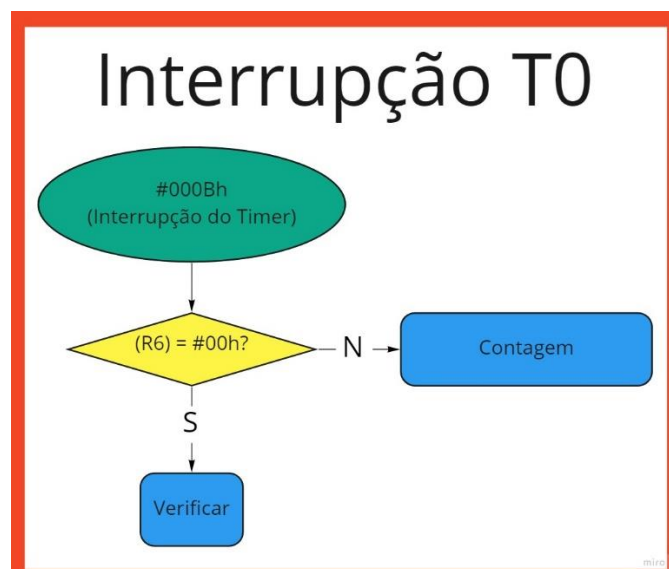
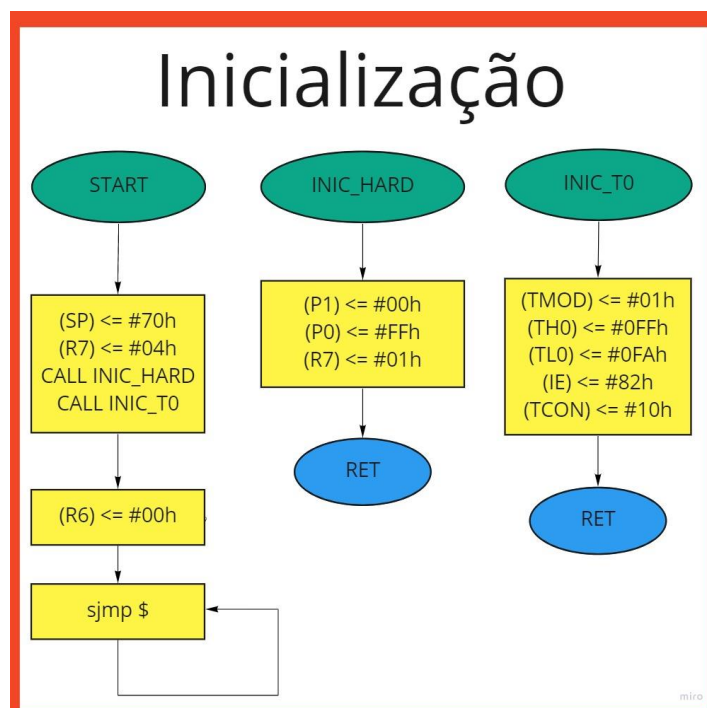


Cada um dos 8 estados do sistema foi representado por um número, como os registradores armazenam 8 bits, cada estado representa um dos bits ativo.

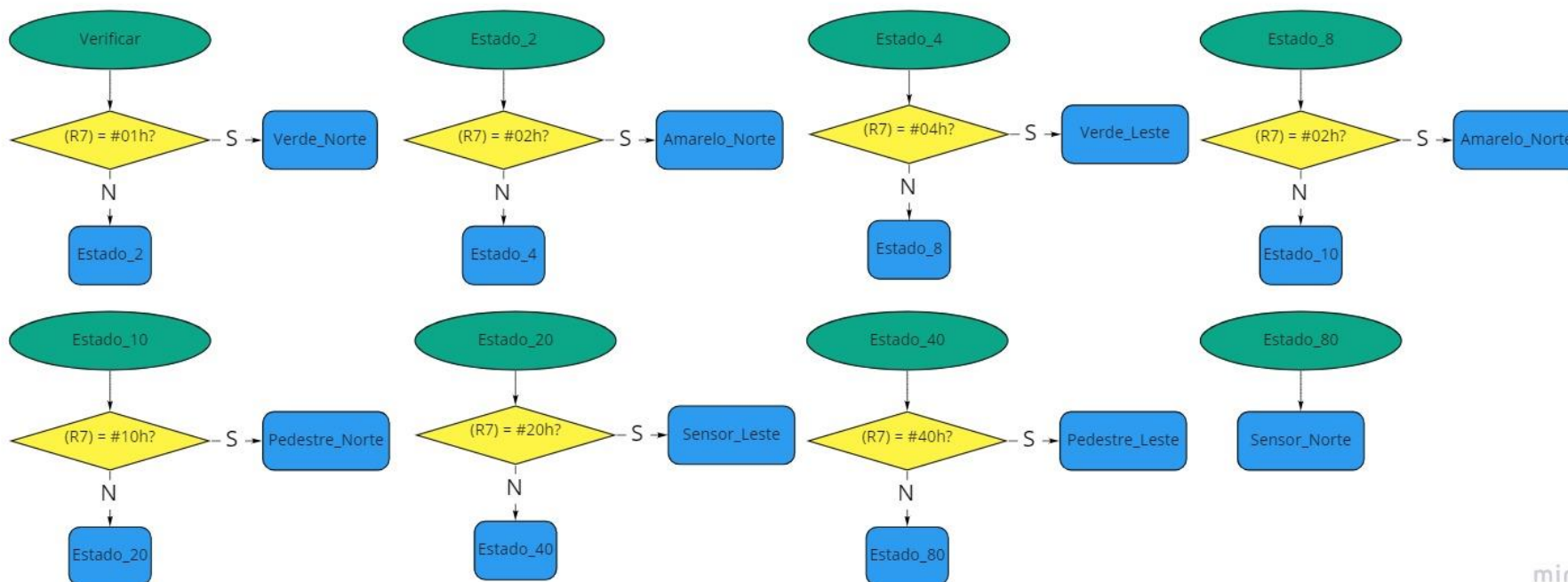
Como visto no diagrama, se não houver sinal dos atuadores nos estados de amarelo, o sistema roda entre os quatro estados principais. Assim os outro quatro estados ficam dispostos para as funções especiais, ou seja, fora do funcionamento padrão.

Os registradores utilizados na lógica foram o R5, R6 e R7. Onde o R5 tem a função de receber o valor do contador correspondente para funcionamento correto do Timer, o R6 tem a função de contar os ciclos de contagem do Timer e controlar a quantidade de tempo, e por fim o R7 tem a função de guardar os valores dos estados.

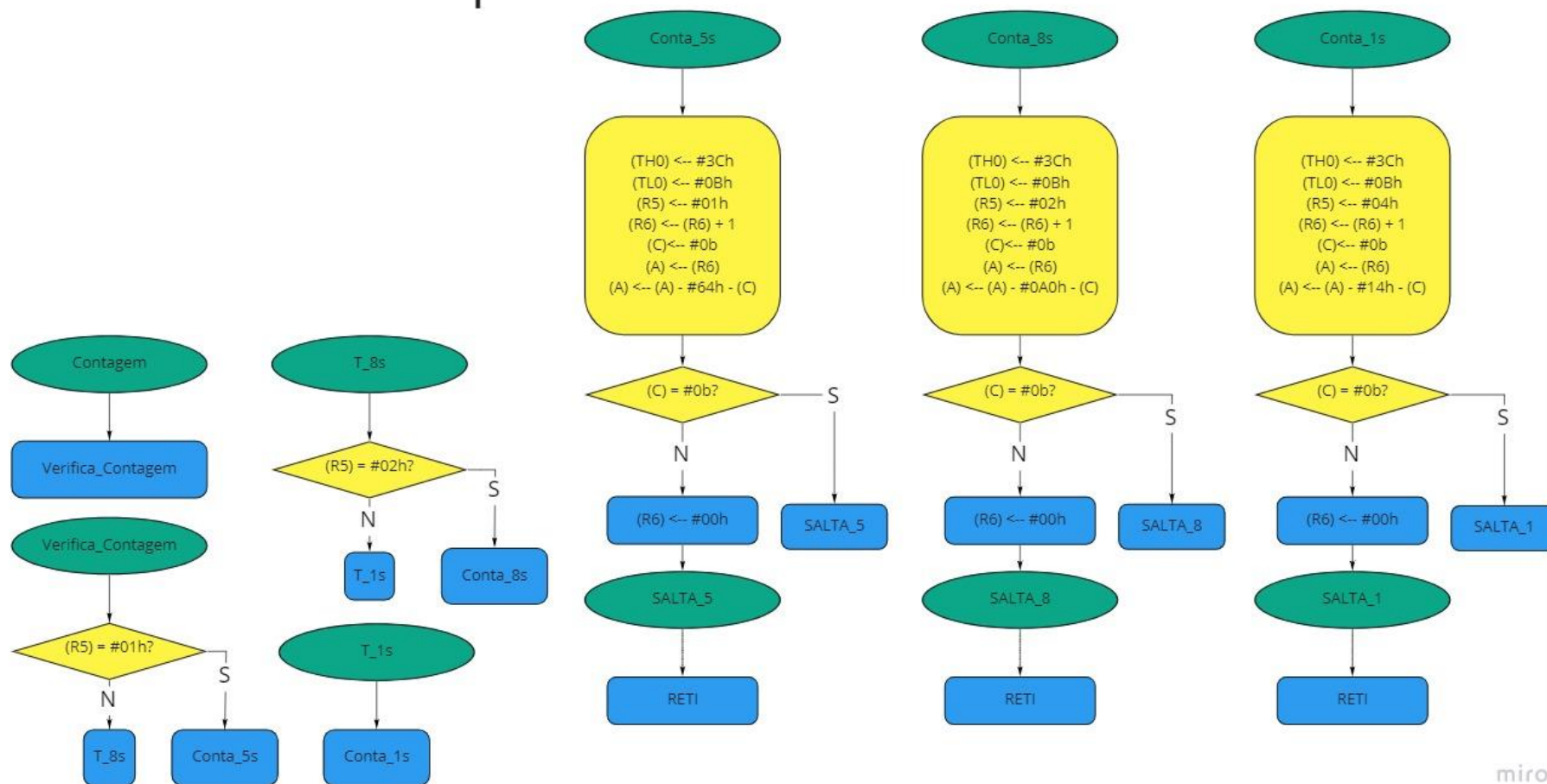
Abaixo está representado o fluxograma do projeto que detalha cada parte da lógica.



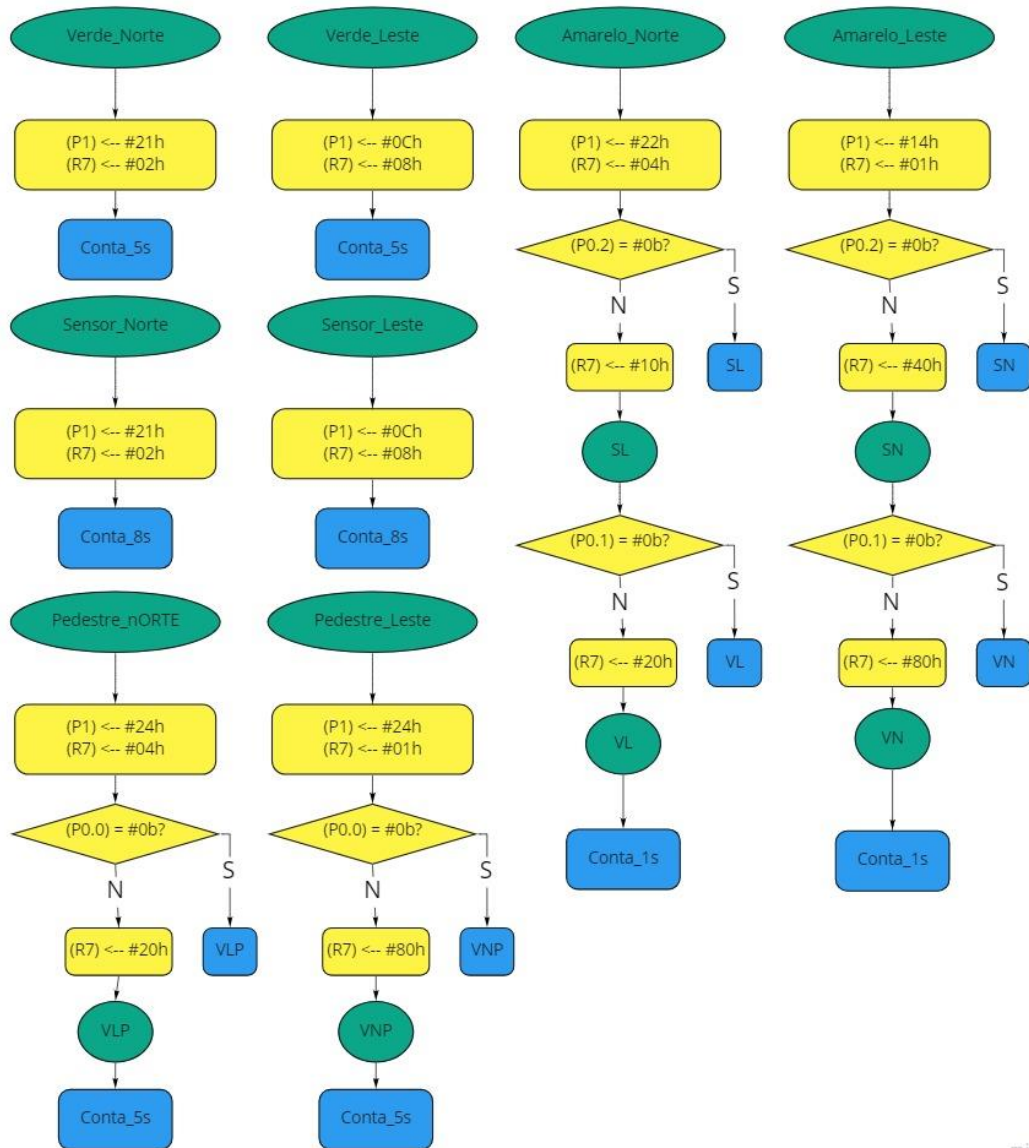
Controle de estados



Controle de tempo



Estados



Para desenvolver a lógica do projeto, todas as etapas foram divididas em setores como representado no fluxograma. Assim conseguimos ter um controle separado de cada etapa da execução e manipular mais facilmente todas as mudanças e adições da lógica.

Primeiramente, o sistema é inicializado deixando prontos todos os recursos importantes para seu funcionamento. Aqui foram configurados os registradores e o Timer, então o sistema deve rodar apenas uma vez essa parte.

A partir desse momento o sistema funciona unicamente baseado nas interrupções geradas pelo Timer 0, ou seja, salta para o endereço 000Bh sempre que o contador do Timer estoura. Nessa parte o sistema deve escolher para qual verificação ele deve ir, se caso a contagem de tempo estiver inativa ele vai para o controle de estados, se estiver ativa vai para o controle de tempo.

No controle de estados, o sistema deve selecionar o estado no qual ele deve entrar baseado no valor do registrador R7, assim como no controle de tempo o sistema deve voltar para a contagem de tempo correta baseado no valor do registrador R5.

Em cada estado do sistema, os LEDs do PORT1 são ligados de acordo com a definição dada no início do projeto e é colocado no R7 o valor do próximo estado. No caso dos estados de amarelo ou pedestre, há uma verificação dos bits do PORT0 usados como chaves para realizar a decisão de qual será o próximo estado, ou seja, qual deve ser o valor a ser carregado no R7.

No controle de tempo, cada uma das três contagens funciona igualmente, pois a única mudança é a quantidade de ciclos que devem ser contados, e claro o valor carregado em R5 para identificar a voltar para iniciar o novo ciclo.

Assim o sistema funciona perfeitamente para esta aplicação, permanece trocando entre os quatro estados principais se não há intervenção de atuadores, e caso haja, serão identificadas as interações pedidas nos estados de amarelo e o sistema segue o caminho determinado para cada situação.

SIMULAÇÃO

Para realizar a simulação e a demonstração do projeto, a lógica foi implementada no Ride7 que também foi utilizado para realizar a simulação via debug.

Abaixo está representado o código fonte do projeto:

```
;Projeto de Microcontroladores (Semáforo com sensores e botão para pedestre)  
$include(REG51.inc)
```

```
code at 0000h  
    ljmp START  
code  
code at 000Bh  
    CJNE R6, #00h, Contagem  
    LJMP Verificar  
Contagem:  
    LJMP Verifica_Contagem  
code  
code at 0500h  
Verifica_Contagem:  
    CJNE R5, #01h, T_8s  
    LJMP Conta_5s  
T_8s:  
    CJNE R5, #02h, T_1s  
    LJMP Conta_8s  
T_1s:  
    LJMP Conta_1s  
code  
code at 0300h  
Verificar:  
    CJNE R7, #01h, Estado_2  
    LJMP Verde_Norte  
Estado_2:  
    CJNE R7, #02h, Estado_4  
    LJMP Amarelo_Norte  
Estado_4:  
    CJNE R7, #04h, Estado_8  
    LJMP Verde_Leste  
Estado_8:  
    CJNE R7, #08h, Estado_10  
    LJMP Amarelo_Leste  
Estado_10:  
    CJNE R7, #10h, Estado_20  
    LJMP Pedestre_Norte  
Estado_20:  
    CJNE R7, #20h, Estado_40  
    LJMP Sensor_Leste
```

```

Estado_40:
    CJNE R7, #40h, Estado_80
    LJMP Pedestre_Leste
Estado_80:
    CJNE R7, #80h, Estado_4
    LJMP Sensor_Norte
code
code at 0100h
INIC_HARD:    ; Inicia o sistema
    MOV P1, #00h
    MOV P0, #0FFh
    MOV R7, #01h
    RET
INIC_T0:
    MOV TMOD, #01h
    MOV TH0, #0FFh
    MOV TL0, #0FAh
    MOV IE, #82h
    MOV TCON, #10h
    RET
code
code at 0200h
Conta_5s:
    MOV TH0, #3Ch
    MOV TL0, #0Bh
    MOV R5, #01h
    INC R6
    CLR C
    MOV A, R6
    SUBB A, #64h
    JC SALTA_5
    MOV R6, #00h
SALTA_5: reti

Conta_8s:
    MOV TH0, #3Ch
    MOV TL0, #0Bh
    MOV R5, #02h
    INC R6
    CLR C
    MOV A, R6
    SUBB A, #0A0h
    JC SALTA_8
    MOV R6, #00h
SALTA_8: reti

Conta_1s:
    MOV TH0, #3Ch
    MOV TL0, #0Bh
    MOV R5, #04h
    INC R6
    CLR C
    MOV A, R6
    SUBB A, #14h
    JC SALTA_1

```

MOV R6, #00h
SALTA_1: reti

Verde_Norte: ; Acende o bit 0 do PORT1 por 5 segundos (Estado 01h)
MOV P1, #21h
MOV R7, #02h
sjmp Conta_5s

Amarelo_Norte: ; Acende o bit 1 do PORT1 por 1 segundos (Estado 02h)
MOV P1, #22h
MOV R7, #04h
JB P0.2, SL
MOV R7, #10h
SL: JB P0.1, VL
MOV R7, #20h
VL: sjmp Conta_1s

Sensor_Norte: ; Acende o bit 0 do PORT1 por 8 segundos (Estado 80h)
MOV P1, #21h
MOV R7, #02h
sjmp Conta_8s

Verde_Leste: ; Acende o bit 3 do PORT1 por 5 segundos (Estado 04h)
MOV P1, #0Ch
MOV R7, #08h
sjmp Conta_5s

Amarelo_Leste: ; Acende o bit 4 do PORT1 por 1 segundos (Estado 08h)
MOV P1, #14h
MOV R7, #01h
JB P0.2, SN
MOV R7, #40h
SN: JB P0.0, VN
MOV R7, #80h
VN: sjmp Conta_1s

Sensor_Leste: ; Acende o bit 3 do PORT1 por 8 segundos (Estado 20h)
MOV P1, #0Ch
MOV R7, #08h
sjmp Conta_8s

Pedestre_Norte: ; Acende os bits 2 e 5 do PORT1 por 5 segundos (Estado 10h)
MOV P1, #24h
MOV R7, #04h
JB P0.1, VLP
MOV R7, #20h
VLP: ljmp Conta_5s

Pedestre_Leste: ; Acende os bits 2 e 5 do PORT1 por 5 segundos (Estado 40h)
MOV P1, #24h
MOV R7, #01h
JB P0.0, VNP
MOV R7, #80h
VNP: ljmp Conta_5s
code

```
code at 0033h
START:
    MOV SP, #70h
    lcall INIC_HARD
    lcall INIC_T0
    MOV R6, #00h
    sjmp $
end
code
END
```

Abaixo estão algumas telas da simulação feita no Ride, onde representam cada estado.

Início:

Debug

Projeto_Semáforo.a51

Code...

```
2 $include(REG51.inc)
3
4 code at 0000h
5     ljmp START
6 code
7 code at 000Bh
8     CJNE R6, #00h, Contagem
9     LJMP Verificar
10 Contagem:
11     LJMP Verifica_Contagem
12 code
13 code at 0500h
14 Verifica_Contagem:
15     CJNE R5, #01h, T_8s
16     LJMP Conta_5s
```

Port 1 [Semáforo]

LATCH 00

0 No Connection

1 No Connection

2 No Connection

3 No Connection

4 No Connection

5 No Connection

6 No Connection

7 No Connection

Port 2 [Semáforo]

LATCH FF

0 No Connection

1 No Connection

2 No Connection

3 No Connection

4 No Connection

5 No Connection

6 No Connection

7 No Connection

Port 0 [Semáforo]

LATCH FF

0 No Connection

1 No Connection

2 No Connection

3 No Connection

4 No Connection

5 No Connection

6 No Connection

7 No Connection

Main Registers [Semáforo]

CPU	Bank	Data	Hardware
PC	000E	R8 00	@R0 00 P0 FF
ACC	00	R0 00	@R1 00 P1 00
PSW	00	R1 00	@DPTR FF P2 FF
SP	72	R2 00	X@R0 FF P3 FF
DPTR	0000	R3 00	X@R1 FF TCON 10
B	00	R4 00	SPX XX THL0 0007
C	0	R5 00	BANK XX THL1 0000
EA	1	R6 00	Task XX THL2 0000
IE	82	R7 01	TaskP XX PCON 00

Disassembly View [Semáforo]

search symbol

Hex

Address	Symbol	Code	Mnemonic
000A:		FF	db 0FFh
			CJNE R6, #00h, Contagem
000B:		BE0003	CJNE R6, #00h, CONTAGEM
			LJMP Verificar
000E:		020300	LJMP VERIFICAR (goto: 300H)
			LJMP Verifica_Contagem
0011:	CONTAGEM	020500	LJMP VERIFICA_CONTAGEM
0014:		FF	db 0FFh
0015:		FF	db 0FFh
0016:		FF	db 0FFh
0017:		FF	db 0FFh

Sfr View [Semáforo]

search symbol

8

Ready

Verde Norte:

Debug

Projeto_Semáforo.a51

Code...

```
89      MOV TH0, #3Ch
90      MOV TLO, #0Bh
91      MOV R5, #04h
92      INC R6
93      CLR C
94      MOV A, R6
95      SUBB A, #14h
96      JC SALTA_1
97      MOV R6, #00h
98      SALTA_1: reti
99
100     Verde_Norte: ; Acende o bit 0 d
101      MOV P1, #21h
102      MOV R7, #02h
103      sjmp Conta_5s
```

Port 1 [Semáforo]

LATCH 21

Port 2 [Semáforo]

LATCH FF

Main Registers [Semáforo]

CPU	Bank	Data	Hardware
PC	023B	R8 00	@R0 00 P0 FF
ACC	00	R0 00	@R1 00 P1 21
PSW	00	R1 00	@DPTR FF P2 FF
SP	72	R2 00	X@R0 FF P3 FF
DPTR	0000	R3 00	X@R1 FF TCON 10
B	00	R4 00	SPX XX THL0 0010
C	0	R5 00	BANK XX THL1 0000
EA	1	R6 00	Task XX THL2 0000
IE	82	R7 02	TaskP XX PCON 00

Disassembly View [Semáforo]

search symbol

Hex

Address	Symbol	Code	Mnemonic
0236:	VERDE_NORTE	759021	MOV P1, #21h
	MOV R7, #02h		
0239:		7F02	MOV R7, #02h
	sjmp Conta_5s		
023B:		80C3	SJMP CONTA_5S (goto: 200H)
	MOV P1, #22h		
023D:	AMARELO_NORTE	759022	MOV P1, #22h
	MOV R7, #04h		
0240:		7F04	MOV R7, #04h
	JB P0.2, SL		
0242:		208202	JB 82h, SL
	MOV R7, #10h		

Port 0 [Semáforo]

LATCH FF

Sfr View [Semáforo]

search symbol

8

Ready

0s.000ms.045 (45 cycles) 103:1 CAP NUM INS

Amarelo Norte:

[illegible]

Pedestre Norte:

Debug

Projeto_Semáforo.a51

Code...

```
129 SN: JB P0.0, VN
130 MOV R7, #80h
131 VN: sjmp Conta_1s
132
133 Sensor_Leste: ; Acende o bit 3 d
134 MOV P1, #0Ch
135 MOV R7, #08h
136 sjmp Conta_8s
137
138 Pedestre_Norte: ; Acende os bits 2
139 MOV P1, #24h
140 MOV R7, #04h
141 JB P0.1, VLP
142 MOV R7, #20h
143 VLP: ljmp Conta_5s
```

Port 1 [Semáforo]

LATCH 24

0 No Connection
1 No Connection
2 No Connection
3 No Connection
4 No Connection
5 No Connection
6 No Connection
7 No Connection

Port 2 [Semáforo]

LATCH FF

0 No Connection
1 No Connection
2 No Connection
3 No Connection
4 No Connection
5 No Connection
6 No Connection
7 No Connection

Port 0 [Semáforo]

LATCH FF

0 Ground
1 Ground
2 Ground
3 No Connection
4 No Connection
5 No Connection
6 No Connection
7 No Connection

Main Registers [Semáforo]

CPU	Bank	Data	Hardware
PC	027E	R8 00	@R0 P0 FF
ACC	00	R0 00	@R1 P1 24
PSW	00	R1 00	@DPTR P2 FF
SP	72	R2 00	X@R0 P3 FF
DPTR	0000	R3 00	X@R1 TCON 10
B	00	R4 00	SPX THL0 001B
C	0	R5 04	BANK THL1 0000
EA	1	R6 00	Task THL2 0000
IE	82	R7 20	TaskP PCON 00

Disassembly View [Semáforo]

search symbol

Hex

Address	Symbol	Code	Mnemonic
0279:		208102	JB 81h, VLP
	MOV R7, #20h		
027C:		7F20	MOV R7, #20h
	VLP: ljmp Conta_5s		
027E:	VLP	020200	LJMP CONTA_5S (goto: 200H)
	MOV P1, #24h		
0281:	PEDESTRE_LESTE	759024	MOV P1, #24h
	MOV R7, #01h		
0284:		7F01	MOV R7, #01h
	JB P0.0, VNP		
0286:		208002	JB 80h, VNP
	MOV R7, #80h		

Sfr View [Semáforo]

search symbol

8

80:	FF	72	00	00	00	00	00	00	00	00	00	00	00	00	00
88:	10	01	1B	00	00	00	00	00	00	00	00	00	00	00	00
90:	24	00	00	00	00	00	00	00	00	00	00	00	00	00	00
98:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A0:	FF	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A8:	82	00	00	00	00	00	00	00	00	00	00	00	00	00	00
B0:	FF	00	00	00	00	00	00	00	00	00	00	00	00	00	00
B8:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
C0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
C8:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D8:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
E0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
E8:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
F0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
F8:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Ready

Foreward

00:00:12.043ms.185 (12.043.185 cycles) 143:1 CAP NUM INS

Sensor Leste:

The screenshot displays the Proteus 8.09 IDE interface for the 'Projeto_Semáforo.a51' project. The main window shows the assembly code for the 'Semaforo' component, with the following instructions visible:

```

122      sjmp Conta_5s
123
124 Amarelo_Leste: ; Acende o bit 4 d
125      MOV P1, #14h
126      MOV R7, #01h
127      JB P0.2, SN
128      MOV R7, #40h
129      SN:    JB P0.0, VN
130      MOV R7, #80h
131      VN:    sjmp Conta_ls
132
133 Sensor_Leste: ; Acende o bit 3 d
134      MOV P1, #0Ch
135      MOV R7, #08h
136      sjmp Conta_8s

```

The left sidebar shows the project structure, including 'Data Dump', 'Disassembly View', 'Code View', 'Xdata View', 'Data View', 'Sfr View', 'Bit View', 'Main Registers', 'Peripherals', 'Interrupt Control', 'Timer 0', 'Timer 1', 'Port 0', and 'Port 1'. The right sidebar shows the 'Main Registers [Semaforo]' table, the 'Sfr View [Semaforo]' table, and the 'Port 0 [Semaforo]', 'Port 1 [Semaforo]', and 'Port 2 [Semaforo]' pin configurations.

The 'Main Registers [Semaforo]' table is as follows:

CPU	Bank	Data	Hardware
PC	0272	R8 00	@R0 00 P0 FF
ACC	00	R0 00	@R1 00 P1 0C
PSW	00	R1 00	@DPTR FF P2 FF
SP	72	R2 00	X@R0 FF P3 FF
DPTR	0000	R3 00	X@R1 FF TCON 10
B	00	R4 00	SPX XX THL0 001A
C	0	R5 01	BANK XX THL1 0000
EA	1	R6 00	Task THL2 0000
IE	82	R7 08	TaskP XX PCON 00

The 'Sfr View [Semaforo]' table is as follows:

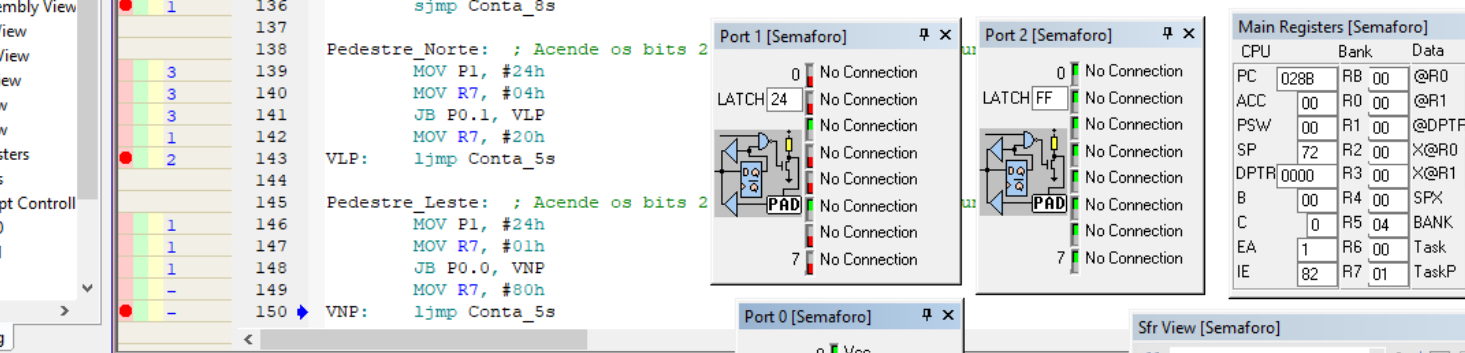
Address	Symbol	Code	Mnemonic
026D:	SENSOR_LESTE	75900C	MOV P1, #0Ch
0270:		7F08	MOV R7, #08h
0272:		809E	SJMP CONTA_8S (goto: 212H)
0274:	PEDESTRE_NORTE	759024	MOV P1, #24h
0277:		7F04	MOV R7, #04h
0279:		208102	JB 81h, VLP

The bottom status bar shows 'Ready' and '00:00:17.061ms.101 (17.061.101 cycles)'.

Amarelo Leste:

[illegible]

Pedestre Leste:



Sensor Norte:

Debug

Projeto_Semáforo.a51

Code...

2 103 sjmp Conta_5s

104

105 Amarelo_Norte: ; Acende o bit 1 d

106 MOV P1, #22h

107 MOV R7, #04h

108 JB P0.2, SL

109 MOV R7, #10h

110 SL: JB P0.1, VL

111 MOV R7, #20h

112 VL: sjmp Conta_1s

113

114 Sensor_Norte: ; Acende o bit 0 d

115 MOV P1, #21h

116 MOV R7, #02h

117 sjmp Conta_8s

Port 1 [Semáforo]

LATCH 21

0 No Connection

No Connection

No Connection

No Connection

No Connection

No Connection

No Connection

No Connection

7 No Connection

Port 2 [Semáforo]

LATCH FF

0 No Connection

No Connection

No Connection

No Connection

No Connection

No Connection

No Connection

No Connection

7 No Connection

Port 0 [Semáforo]

LATCH FF

0 Ground

Ground

Ground

No Connection

No Connection

No Connection

No Connection

No Connection

7 No Connection

Main Registers [Semáforo]

CPU	Bank	Data	Hardware
PC	0253	R8 00	@R0 00 P0 FF
ACC	00	R0 00	@R1 00 P1 21
PSW	00	R1 00	@DPTR FF P2 FF
SP	72	R2 00	X@R0 FF P3 FF
DPTR	0000	R3 00	X@R1 FF TCON 10
B	00	R4 00	SPX THL0 001E
C	0	R5 04	BANK THL1 0000
EA	1	R6 00	Task THL2 0000
IE	82	R7 02	TaskP PCON 00

Disassembly View [Semáforo]

search symbol

Address Symbol Code Mnemonic

024E: SENSOR_NORTE 759021 MOV P1, #21h

MOV R7, #02h

0251: 7F02 MOV R7, #02h

sjmp Conta_8s

0253: 80BD SJMP CONTA_8S (goto: 212H)

MOV P1, #0Ch

0255: VERDE_LESTE 75900C MOV P1, #0Ch

MOV R7, #08h

0258: 7F08 MOV R7, #08h

sjmp Conta_5s

025A: 80A4 SJMP CONTA_5S

MOV P1, #14h

Build Log Disassembly View [Semáforo]

Sfr View [Semáforo]

search symbol

80: FF 72 00 00 00 00 00 00

88: 10 01 1E 00 00 00 00 00

90: 21 00 00 00 00 00 00 00 !

98: 00 00 00 00 00 00 00 00

A0: FF 00 00 00 00 00 00 00

A8: 82 00 00 00 00 00 00 00

B0: FF 00 00 00 00 00 00 00

B8: 00 00 00 00 00 00 00 00

C0: 00 00 00 00 00 00 00 00

C8: 00 00 00 00 00 00 00 00

D0: 00 00 00 00 00 00 00 00

D8: 00 00 00 00 00 00 00 00

E0: 00 00 00 00 00 00 00 00

E8: 00 00 00 00 00 00 00 00

F0: 00 00 00 00 00 00 00 00

F8: 00 00 00 00 00 00 00 00

Ready

Verde Leste:

Debug

Projeto_Semáforo.a51

Code...

```
108 JB P0.2, SL
109 MOV R7, #10h
110 SL: JB P0.1, VL
111 MOV R7, #20h
112 VL: sjmp Conta_1s
113
114 Sensor_Norte: ; Acende o bit 0 d
115 MOV P1, #21h
116 MOV R7, #02h
117 sjmp Conta_8s
118
119 Verde_Leste: ; Acende o bit 3 d
120 MOV P1, #0Ch
121 MOV R7, #08h
122 sjmp Conta_5s
```

Port 1 [Semáforo]

Port 2 [Semáforo]

Main Registers [Semáforo]

CPU	Bank	Data	Hardware
PC	025A	R8 00 @R0	P0 FF
ACC	00	R0 00 @R1	P1 0C
PSW	00	R1 00 @DPTR	P2 FF
SP	72	R2 00 X@R0	P3 FF
DPTR	0000	R3 00 X@R1	TCON 10
B	00	R4 00 SPX	THL0 0014
C	0	R5 01 BANK	THL1 0000
EA	1	R6 00 Task	THL2 0000
IE	82	R7 08 TaskP	PCON 00

Port 0 [Semáforo]

Sfr View [Semáforo]

search symbol

Address	Symbol	Code	Mnemonic
0255:	VERDE_LESTE	75900C	MOV P1, #0Ch
0258:		7F08	MOV R7, #08h
025A:		80A4	SJMP CONTA_5S (goto: 200H)
025C:	AMARELO_LESTE	759014	MOV P1, #14h
025F:		7F01	MOV R7, #01h
0261:		208202	JB 82h, SN

Build Log

Disassembly View [Semáforo]

Ready

CONCLUSÃO E CONSIDERAÇÕES

Desenvolver um projeto do zero é sempre um desafio, mas também muito enriquecedor. Com esse sistema de semáforo, pudemos aplicar praticamente todas as ferramentas aprendidas no curso, e isso além de fixar bem o conteúdo faz com que tenhamos também um aprimoramento na lógica e trabalho em grupo.

Nosso grupo gostou bastante do tema desenvolvido e então pensamos em algumas melhorias e recursos novos. Poderiam ser adicionados displays de contagem de tempo, mais estágios de identificação de atuadores ou também um sistema mais inteligente para controle de tempo podendo disponibilizar mais variedade de acordo com a demanda de carros.

Nós do grupo agradecemos acima de tudo ao Professor Orientador André Luiz Perin pelas ótimas aulas e pelas dicas para o projeto, com certeza foi de extrema importância para o desenvolvimento do curso e do projeto, tanto quanto para nossa formação.