



UNIVERSIDADE ESTADUAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
BACHARELADO EM CIÊNCIAS
DA COMPUTAÇÃO

LUCAS DE LUCENA SIQUEIRA

DISCIPLINA: LABORATÓRIO DE ORGANIZAÇÃO E ARQUITETURA DE
COMPUTADORES

EXPERIMENTO 02

RELATÓRIO - ROTAÇÃO DE LED'S

CAMPINA GRANDE - PB

2021

SUMÁRIO

1. RESUMO.....	3
2. INTRODUÇÃO.....	3
3. MATERIAL E MÉTODOS.....	3
3.1. OBJETIVO.....	3
3.2. SOFTWARE NECESSÁRIO.....	3
3.3. HARDWARE NECESSÁRIO.....	3
3.4. PROCEDIMENTOS.....	3
3.4.1. LIGAÇÕES DO PAINEL DE LED.....	4
3.4.2. LISTAGEM DO PROGRAMA EM LM.....	5
3.4.3. FUNCIONAMENTO DO PROGRAMA.....	6
3.4.4. FUNCIONAMENTO DO PAINEL DE LED'S.....	11
3.4.5. CONTEÚDO DO ARQUIVO “.lst”.....	13
4. RESULTADOS E DISCUSSÃO.....	15
5. CONCLUSÕES.....	15
6. REFERÊNCIAS BIBLIOGRÁFICAS.....	15

1. RESUMO

Neste trabalho em questão é possível observar alguns conceitos básicos e importantes sobre a programação do microcontrolador MCS51 em Assembly a partir do simulador MCU8051 IDE. Foi possível absorver algumas noções que dizem respeito à rotação de led 's com a utilização do simulador e dos comandos da linguagem Assembly.

2. INTRODUÇÃO

O presente trabalho tem como objetivo introduzir o estudo do microcontrolador MCS51 a partir do uso do simulador MCU8051 IDE. Tendo o Assembly como linguagem dominante na programação do referente microcontrolador, se faz necessário também o estudo da mesma a partir de consultas em seu *Datasheet*, que é nada mais do que uma folha com dados e especificações técnicas e de desempenho do produto levado em consideração, que no nosso caso é o microcontrolador MCS51.

3. MATERIAL E MÉTODOS

3.1. OBJETIVO

O experimento em questão tem como objetivo induzir o estudo e a prática da programação do microcontrolador MCS51 a partir do simulador MCU8051 IDE com o uso da linguagem Assembly. Tornando possível estudar a rotação de led 's no simulador.

3.2. HARDWARE NECESSÁRIO

- Computador com sistema operacional superior ou equivalente ao Windows 7.

3.3. SOFTWARE NECESSÁRIO

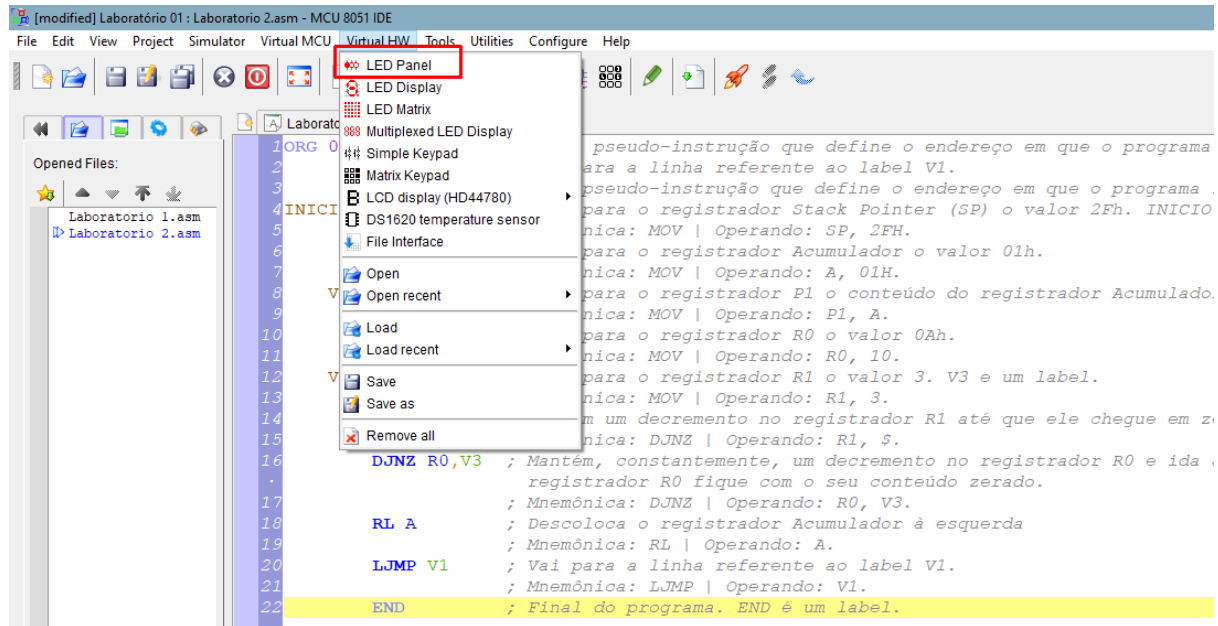
- Simulador MCU 8051 IDE.

3.4. PROCEDIMENTOS

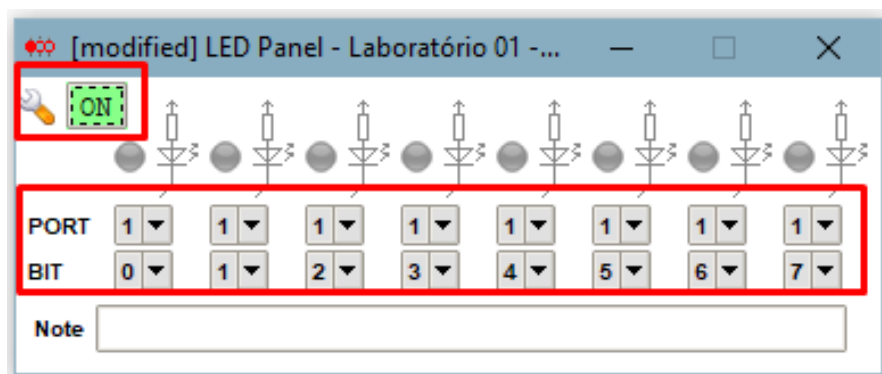
3.4.1. LIGAÇÕES DO PAINEL DE LED

Para iniciar o experimento é necessário fazer a ligação das portas e seus respectivos bits no painel de led's do simulador.

Localização do painel de led's:



Para configurar o painel, basta ativá-lo e relacionar cada um dos oito led's a uma porta e a um bit do microcontrolador como mostra a imagem a seguir:



3.4.2. LISTAGEM DO PROGRAMA EM LM

- ORG 00H

É uma pseudo-instrução que define o endereço em que o programa irá começar ou continuar.

- LJMP INICIO

É um label que tem como função apontar para um determinado trecho de código.

- ORG 30H

É uma pseudo-instrução que define o endereço em que o programa irá começar ou continuar.

- INICIO: MOV SP,#2FH

Move para o registrador SP um determinado valor.

- MOV A,#01H

Move para o registrador Acumulador um determinado valor.

- V1: MOV P1,A

Move para o registrador P1 o conteúdo do registrador Acumulador.

- MOV R0,#10

Move para o registrador R0 um determinado valor.

- V3: MOV R1,#3

Move para o registrador R1 um determinado valor.

- DJNZ R1,\$

Mantém um decremento no registrador R1 até que ele chegue a zero, só assim o programa dará procedência para as próximas linhas de comando.

- DJNZ R0,V3

Mantém, constantemente, um decremento no registrador R0 e ida até à localização do label V3 até que o registrador R0 fique com o seu conteúdo zerado.

- RLA

Desloca o registrador Acumulador à esquerda, movendo todos os bits do Acumulador uma casa à esquerda e adicionando um zero no bit mais à direita.

- LJMP V1

É um label que tem como função apontar para um determinado trecho de código.

- END

Define o final do programa.

3.4.3. FUNCIONAMENTO DO PROGRAMA

Aqui estará presente toda a descrição detalhada de cada linha do programa utilizado como exemplo, apresentando em cada linha seu respectivo funcionamento comentado.

```

1 ORG 00H ; É uma pseudo-instrução que define o endereço em que o programa irá começar.
2 LJMP INICIO ; Vai para a linha referente ao label INICIO.
3 ORG 30H ; É uma pseudo-instrução que define o endereço em que o programa irá continuar.
4 INICIO: MOV SP,#2FH ; Move para o registrador Stack Pointer (SP) o valor 2Fh.
5 MOV A,#01H ; Move para o registrador Acumulador o valor 01h.
6 V1: MOV P1,A ; Move para o registrador P1 o conteúdo do registrador Acumulador.
7 MOV R0,#10 ; Move para o registrador R0 o valor 0Ah.
8 V3: MOV R1,#3 ; Move para o registrador R1 o valor 3.
9 DJNZ R0,V3 ; Mantém um decremento no registrador R0 até que ele chegue em zero para seguir para próxima linha do programa.
10 ; Mantém, constantemente, um decremento no registrador R0 e ida até a localização do label V3 até que o
11 ; registrador R0 fique com o seu conteúdo zerado.
12 RLA ; Desloca o registrador Acumulador à esquerda, movendo todos os bits do Acumulador uma casa à esquerda e
13 ; adicionando um zero no bit mais à direita.
14 LJMP V1 ; Vai para a linha referente ao label V1.
15 END ; Final do programa.

```

Comentários e execução do código acima:

ORG 00H

Função: Indica que o programa vai começar no endereço 0H (PC = 0).

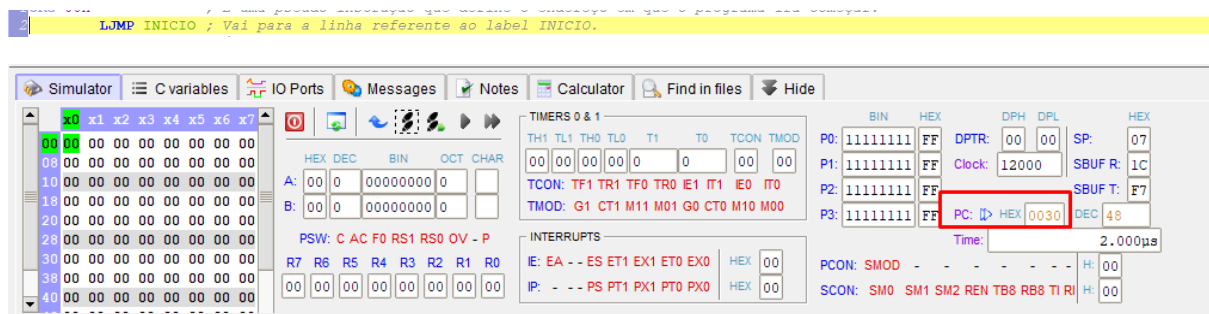
ORG 00H ; É uma pseudo-instrução que define o endereço em que o programa irá começar.

The screenshot shows a microcontroller simulator interface. The PC register is highlighted in red and set to 0000. The interface includes a memory window, registers, and various status indicators.

REG	VALUE
PC	0000
SP	07
P1	FF
P2	FF
P3	FF
R0	00
R1	03
R2	00
R3	00
R4	00
R5	00
R6	00
R7	00

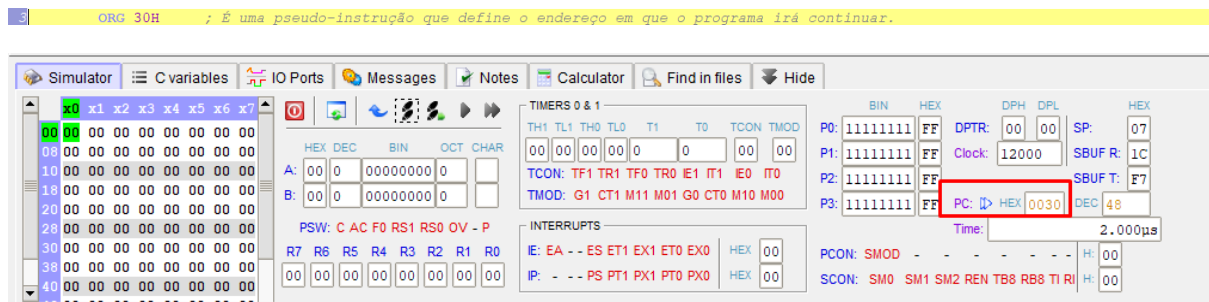
LJMP INICIO

Função: o programa cai para a linha referente ao label INICIO.



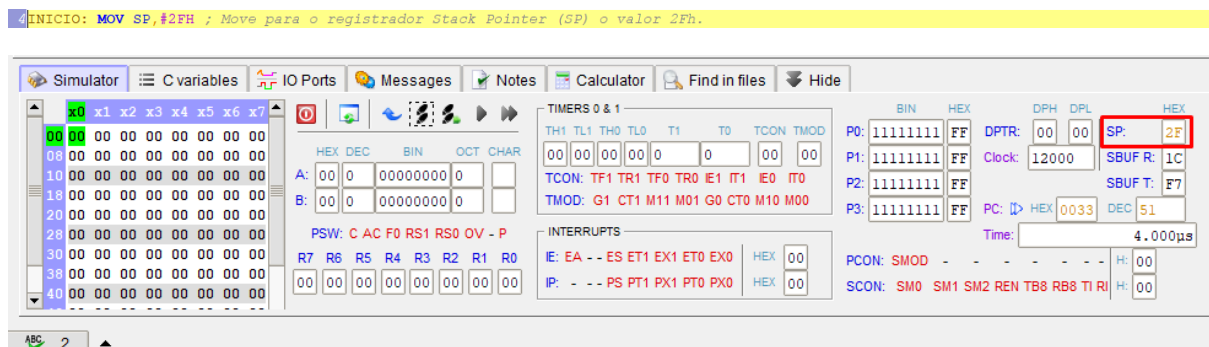
ORG 30H

Função: É uma pseudo-instrução que define o que o programa irá continuar no endereço e memória 30H.



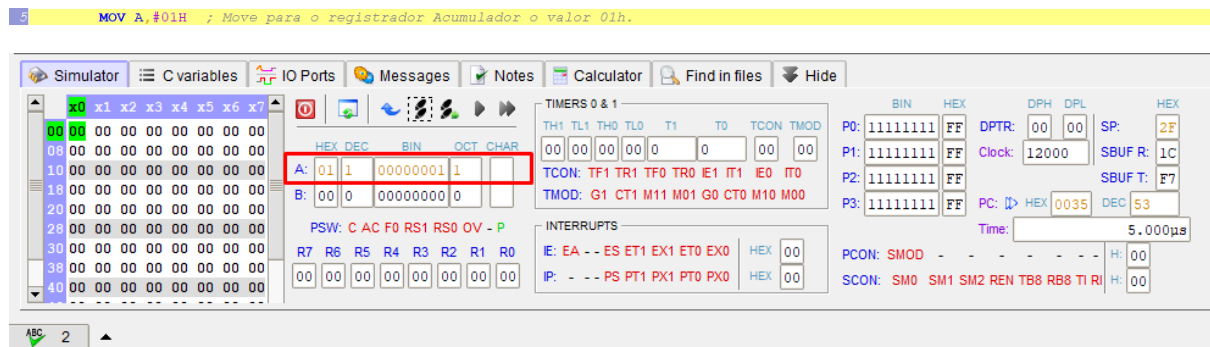
INICIO: MOV SP,#2FH

Função: Move para o registrador Stack Pointer (SP) o valor 2Fh.



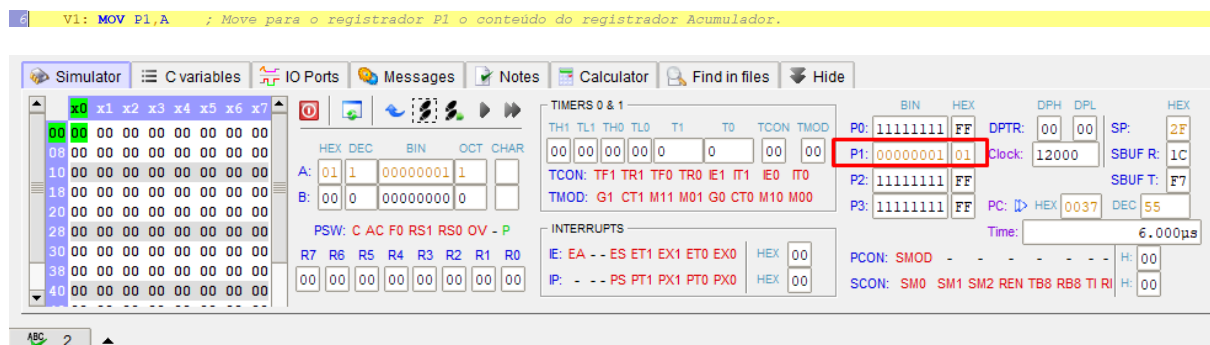
MOV A,#01H

Função: Move para o registrador Acumulador o valor 01H.

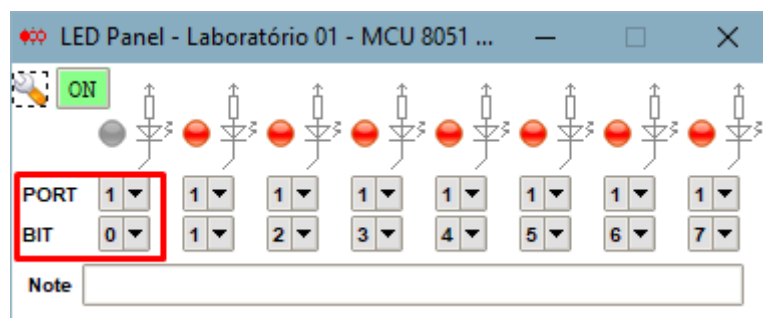


V1: MOV P1,A

Função: Move para o registrador P1 o conteúdo do registrador Acumulador. V1 é um label

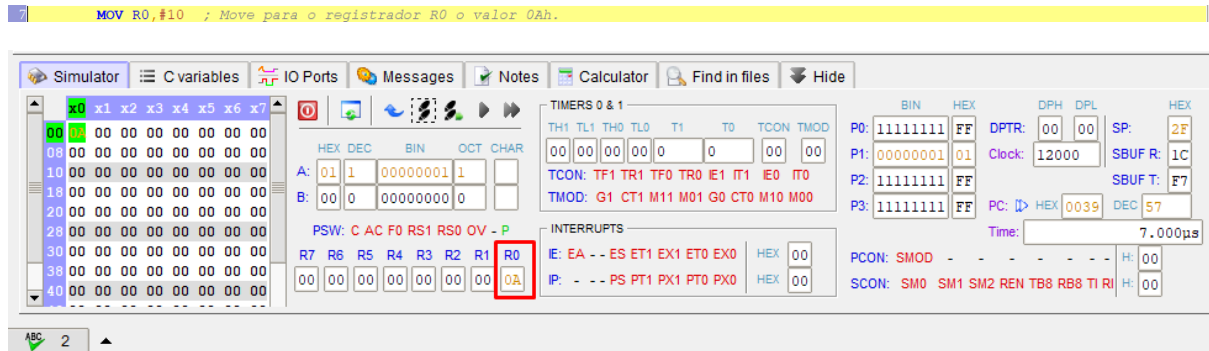


Painel de LED 's: Como o bit 0 do registrador P1 foi alterado para 1, o led associado desligou.



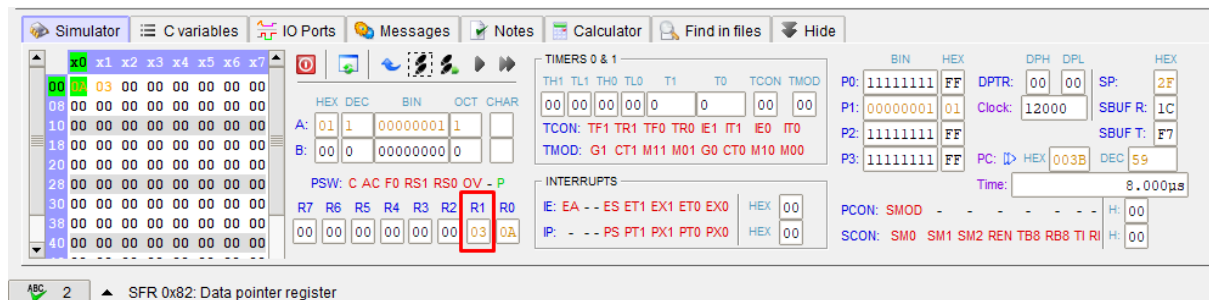
MOV R0,#10

Função: Move para o registrador R0 o valor 0Ah.



V3: MOV R1,#3

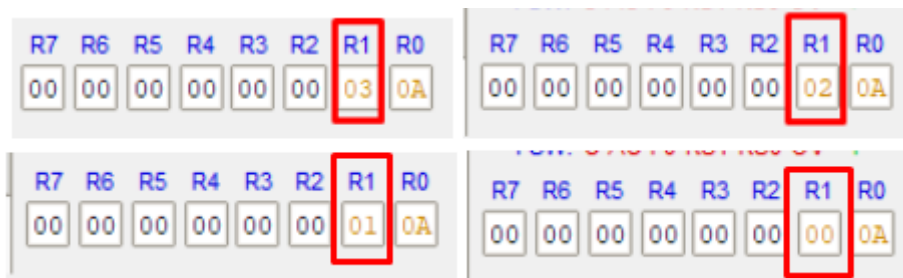
Função: Move para o registrador R1 o valor 3.



DJNZ R1,\$

Função: Mantém um decremento no registrador R1 até que ele chegue em zero para seguir para a próxima linha do programa.

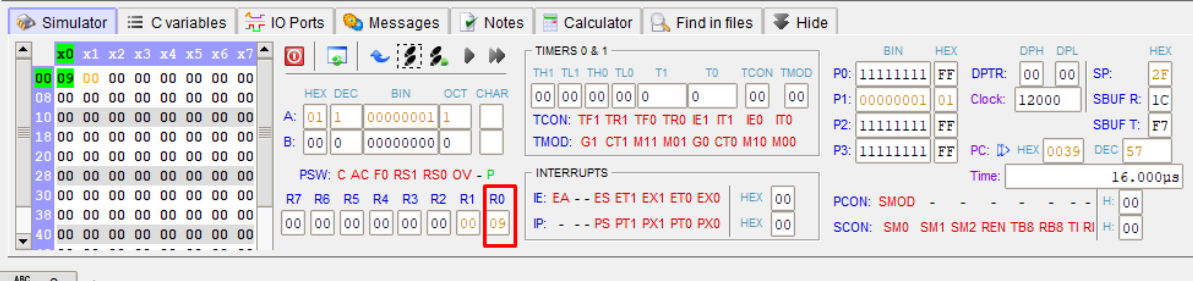
`DJNZ R1,$; Mantém um decremento no registrador R1 até que ele chegue em zero para seguir para próxima linha do programa.`



DJNZ R0,V3

Função: Mantém, constantemente, um decremento no registrador R0 e ida até à localização do label V3 até que o registrador R0 fique com o seu conteúdo zerado.

10 DJNZ R0,V3 ; Mantém, constantemente, um decremento no registrador R0 e ida até à localização do label V3 até que o registrador R0 fique com o seu conteúdo zerado.



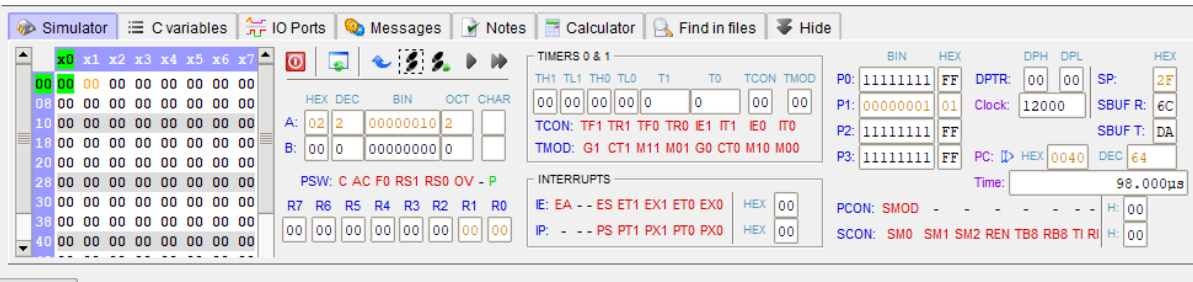
The simulator interface shows the following details:

- Registers:** R0 is highlighted with a red box and contains the value 09. Other registers (R1-R7) contain 00.
- PC (Program Counter):** Contains the value 0039.
- Timers:** T0 and T1 are both 00.
- Interrupts:** IE, IP, and other interrupt-related registers are shown with their respective values.
- Status Bar:** Shows 'ABC 2'.

RLA

Função: Desloca o registrador Acumulador à esquerda, movendo todos os bits do Acumulador uma casa à esquerda e adicionando um zero no bit mais à direita.

11 RLA ; Desloca o registrador Acumulador à esquerda, movendo todos os bits do Acumulador uma casa à esquerda e adicionando um zero no bit mais à direita.



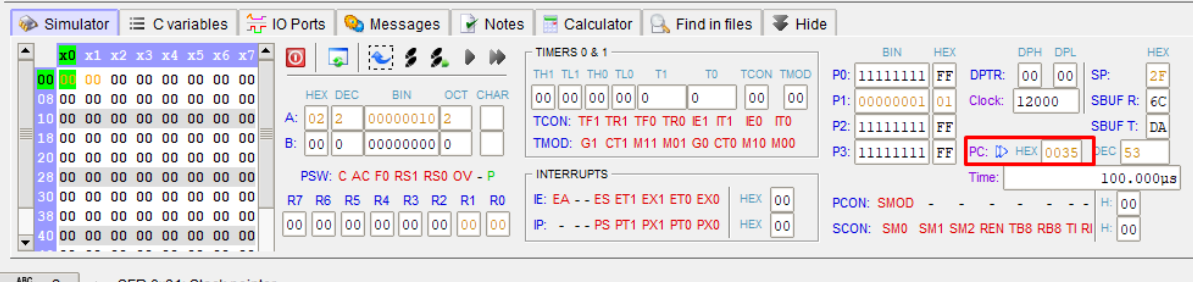
The simulator interface shows the following details:

- Registers:** A (Accumulator) is highlighted with a red box and contains the value 02. Other registers (R1-R7) contain 00.
- PC (Program Counter):** Contains the value 0040.
- Timers:** T0 and T1 are both 00.
- Interrupts:** IE, IP, and other interrupt-related registers are shown with their respective values.
- Status Bar:** Shows 'ABC 2'.

LJMP V1

Função: Vai para a linha referente ao label V1.

12 LJMP V1 ; Vai para a linha referente ao label V1.



The simulator interface shows the following details:

- Registers:** R0-R7 contain 00.
- PC (Program Counter):** Is highlighted with a red box and contains the value 0035.
- Timers:** T0 and T1 are both 00.
- Interrupts:** IE, IP, and other interrupt-related registers are shown with their respective values.
- Status Bar:** Shows 'ABC 2'.

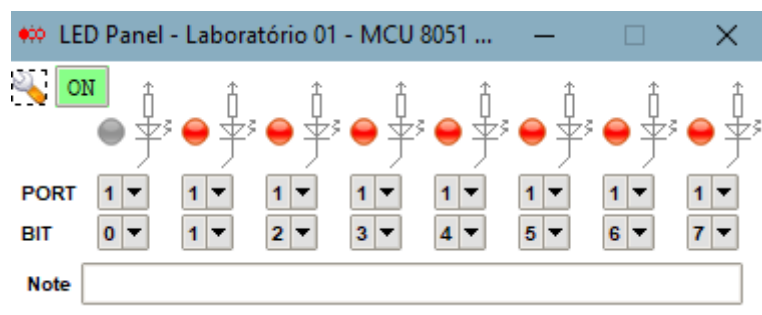
END

Função: Indica o final do programa.

```
15 END ; Final do programa.
```

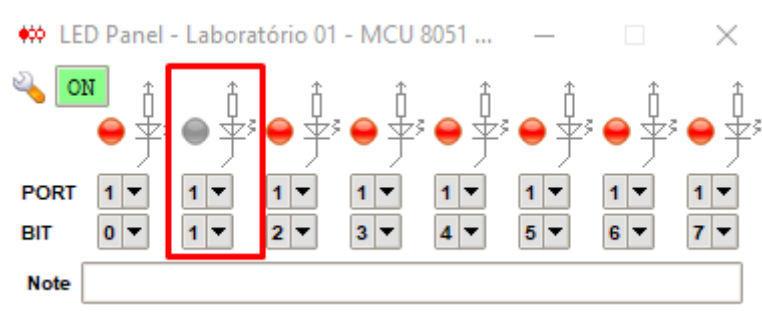
3.4.4. FUNCIONAMENTO DO PAINEL DE LED'S

Inicialmente, o painel tem a primeira alteração quando o programa chega na linha 6 do programa, ficando da seguinte forma:



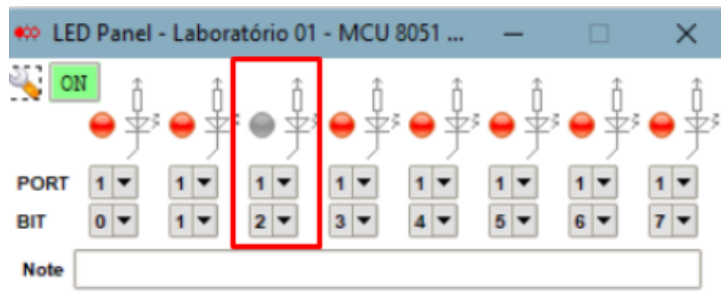
(1º Execução)

Após algumas interações, quando o programa chega na linha 12 que executa um comando referente em voltar para a linha 6 que quando executada, ocorre, consequentemente, a alteração do conteúdo do registrador P1, que fica com o mesmo conteúdo presente no registrador Acumulador (que no início era equivalente a 01H, porém devido ao comando “RL A” presente na linha 11 foi alterado para 02H) há uma alteração no painel de LED's, já que o conteúdo do registrador P1 foi alterado para 00000010, como é mostrado a seguir:

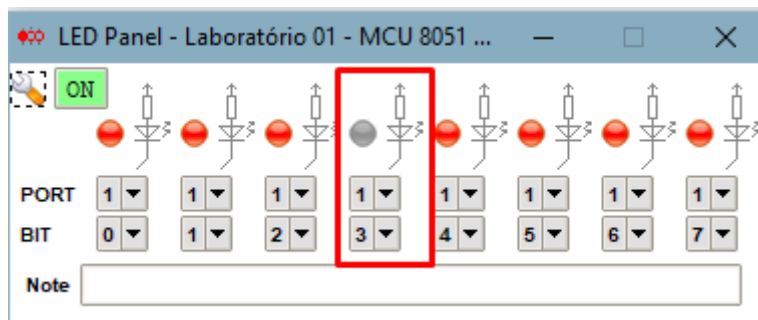


(2º Execução)

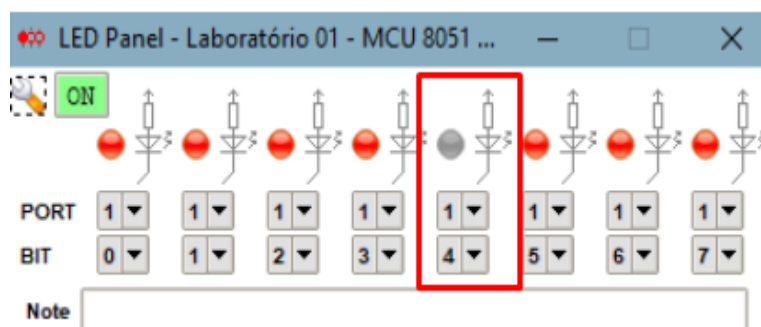
Nota-se que como consequência da listagem do programa em Assembly, sempre haverá essa mudança padronizada no painel de LED 's, em que o led que antes estava desligado ascende e o led referente ao bit seguinte (led à direita) irá desligar, como é mostrado a seguir:



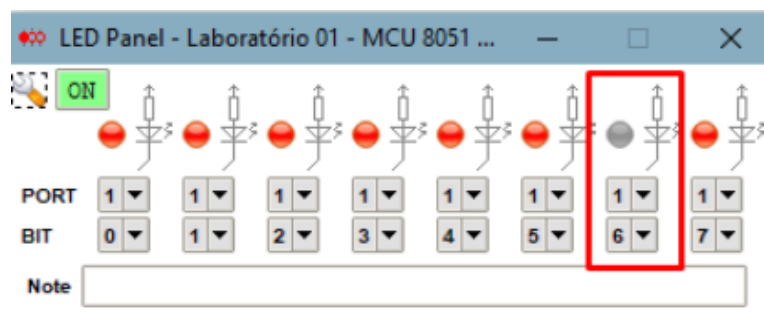
(3° Execução)



(4° Execução)



(5° Execução)



(6° Execução)

[illegible]

Laboratorio 2 - Bloco de Notas					Laboratorio 2 - Bloco de Notas				
Arquivo	Editar	Formatar	Exibir	Ajuda	Arquivo	Editar	Formatar	Exibir	Ajuda
CCF0	B	ADDR	0008H	NOT USED	RS0	B	ADDR	0003H	NOT USED
CCF1	B	ADDR	0009H	NOT USED	RS1	B	ADDR	0004H	NOT USED
CCF2	B	ADDR	000AH	NOT USED	RXD	B	ADDR	000BH	NOT USED
CCF3	B	ADDR	000BH	NOT USED	SADDR	D	ADDR	00A9H	NOT USED
CCF4	B	ADDR	000CH	NOT USED	SADDR_0	D	ADDR	00A9H	NOT USED
CCON	D	ADDR	000BH	NOT USED	SADDR_1	D	ADDR	00AAH	NOT USED
CFINT	C	ADDR	0033H	NOT USED	SADEN	D	ADDR	00B9H	NOT USED
CH	D	ADDR	00F9H	NOT USED	SADEN_0	D	ADDR	00B9H	NOT USED
CKCON	D	ADDR	008FH	NOT USED	SADEN_1	D	ADDR	00BAH	NOT USED
CKCON0	D	ADDR	008FH	NOT USED	SBUF	D	ADDR	0099H	NOT USED
CKRL	D	ADDR	0097H	NOT USED	SCON	D	ADDR	0098H	NOT USED
CKSEL	D	ADDR	0085H	NOT USED	SINT	C	ADDR	0023H	NOT USED
CL	D	ADDR	00E9H	NOT USED	SM0	B	ADDR	009FH	NOT USED
CLKREG	D	ADDR	008FH	NOT USED	SM1	B	ADDR	009EH	NOT USED
CMD	D	ADDR	0009H	NOT USED	SM2	B	ADDR	009DH	NOT USED
CPRL2	B	ADDR	00C8H	NOT USED	SP	D	ADDR	0081H	NOT USED
CR	B	ADDR	00DEH	NOT USED	SPCON	D	ADDR	00C3H	NOT USED
CT2	B	ADDR	00C9H	NOT USED	SPCR	D	ADDR	00D5H	NOT USED
CY	B	ADDR	00D7H	NOT USED	SPDAT	D	ADDR	00C5H	NOT USED
DP0H	D	ADDR	0083H	NOT USED	SPDR	D	ADDR	0086H	NOT USED
DP0L	D	ADDR	0082H	NOT USED	SPSR	D	ADDR	00AAH	NOT USED
DP1H	D	ADDR	0085H	NOT USED	SPSTA	D	ADDR	00C4H	NOT USED
DP1L	D	ADDR	0084H	NOT USED	T0	B	ADDR	00B4H	NOT USED
DPH	D	ADDR	0083H	NOT USED	T1	B	ADDR	00B5H	NOT USED
DPL	D	ADDR	0082H	NOT USED	T2CON	D	ADDR	00C8H	NOT USED
EA	B	ADDR	00AFH	NOT USED	T2MOD	D	ADDR	00C9H	NOT USED
EC	B	ADDR	00AEH	NOT USED	TB8	B	ADDR	009BH	NOT USED
EECON	D	ADDR	0096H	NOT USED	TCLK	B	ADDR	00CCH	NOT USED
ES	B	ADDR	00ACH	NOT USED	TCON	D	ADDR	0088H	NOT USED
ET0	B	ADDR	00A9H	NOT USED	TF0	B	ADDR	008DH	NOT USED
ET1	B	ADDR	00ABH	NOT USED	TF1	B	ADDR	00BFH	NOT USED
ET2	B	ADDR	00ADH	NOT USED	TF2	B	ADDR	00CFH	NOT USED
EX0	B	ADDR	00A8H	NOT USED	TH0	D	ADDR	008CH	NOT USED
EX1	B	ADDR	00AAH	NOT USED	TH1	D	ADDR	008DH	NOT USED
EXEN2	B	ADDR	00CBH	NOT USED	TH2	D	ADDR	00CDH	NOT USED
EXF2	B	ADDR	00CEH	NOT USED	TI	B	ADDR	0099H	NOT USED
EXTI0	C	ADDR	0083H	NOT USED	TIMER0	C	ADDR	000BH	NOT USED
EXTI1	C	ADDR	0013H	NOT USED	TIMER1	C	ADDR	001BH	NOT USED
F0	B	ADDR	00D5H	NOT USED	TIMER2	C	ADDR	002BH	NOT USED
FE	B	ADDR	009FH	NOT USED	TL0	D	ADDR	00BAH	NOT USED
IE	D	ADDR	00A8H	NOT USED	TL1	D	ADDR	00BBH	NOT USED
IE0	B	ADDR	0089H	NOT USED	TL2	D	ADDR	00CCH	NOT USED
IE1	B	ADDR	008BH	NOT USED	TMOD	D	ADDR	00B9H	NOT USED
INITIO	C	ADDR	0030H		TR0	B	ADDR	008CH	NOT USED
INT0	B	ADDR	00B2H	NOT USED	TR1	B	ADDR	008EH	NOT USED
INT1	B	ADDR	00B3H	NOT USED	TR2	B	ADDR	00CAH	NOT USED
IP	D	ADDR	008BH	NOT USED	TXD	B	ADDR	00B1H	NOT USED
IPH	D	ADDR	0087H	NOT USED	V1	C	ADDR	0035H	
IPH0	D	ADDR	0087H	NOT USED	V3	C	ADDR	0039H	
IPH1	D	ADDR	0083H	NOT USED	WDTCON	D	ADDR	00A7H	NOT USED
IPL0	D	ADDR	008BH	NOT USED	WDTPRG	D	ADDR	00A7H	NOT USED
IPL1	D	ADDR	00B2H	NOT USED	WDRST	D	ADDR	00AGH	NOT USED
IT0	B	ADDR	008BH	NOT USED	WR	B	ADDR	00B6H	NOT USED
<					<				

Laboratorio 2 - Bloco de Notas				
Arquivo	Editar	Formatar	Exibir	Ajuda
RS0	B	ADDR	0003H	NOT USED
RS1	B	ADDR	0004H	NOT USED
RXD	B	ADDR	000BH	NOT USED
SADDR	D	ADDR	00A9H	NOT USED
SADDR_0	D	ADDR	00A9H	NOT USED
SADDR_1	D	ADDR	00AAH	NOT USED
SADEN	D	ADDR	00B9H	NOT USED
SADEN_0	D	ADDR	00B9H	NOT USED
SADEN_1	D	ADDR	00BAH	NOT USED
SBUF	D	ADDR	0099H	NOT USED
SCON	D	ADDR	0098H	NOT USED
SINT	C	ADDR	0023H	NOT USED
SM0	B	ADDR	009FH	NOT USED
SM1	B	ADDR	009EH	NOT USED
SM2	B	ADDR	009DH	NOT USED
SP	D	ADDR	0081H	NOT USED
SPCON	D	ADDR	00C3H	NOT USED
SPCR	D	ADDR	00D5H	NOT USED
SPDAT	D	ADDR	00C5H	NOT USED
SPDR	D	ADDR	0086H	NOT USED
SPSR	D	ADDR	00AAH	NOT USED
SPSTA	D	ADDR	00C4H	NOT USED
T0	B	ADDR	00B4H	NOT USED
T1	B	ADDR	00B5H	NOT USED
T2CON	D	ADDR	00C8H	NOT USED
T2MOD	D	ADDR	00C9H	NOT USED
TB8	B	ADDR	009BH	NOT USED
TCLK	B	ADDR	00CCH	NOT USED
TCON	D	ADDR	0088H	NOT USED
TF0	B	ADDR	008DH	NOT USED
TF1	B	ADDR	00BFH	NOT USED
TF2	B	ADDR	00CFH	NOT USED
TH0	D	ADDR	008CH	NOT USED
TH1	D	ADDR	008DH	NOT USED
TH2	D	ADDR	00CDH	NOT USED
TI	B	ADDR	0099H	NOT USED
TIMER0	C	ADDR	000BH	NOT USED
TIMER1	C	ADDR	001BH	NOT USED
TIMER2	C	ADDR	002BH	NOT USED
TL0	D	ADDR	00BAH	NOT USED
TL1	D	ADDR	00BBH	NOT USED
TL2	D	ADDR	00CCH	NOT USED
TMOD	D	ADDR	00B9H	NOT USED
TR0	B	ADDR	008CH	NOT USED
TR1	B	ADDR	008EH	NOT USED
TR2	B	ADDR	00CAH	NOT USED
TXD	B	ADDR	00B1H	NOT USED
V1	C	ADDR	0035H	
V3	C	ADDR	0039H	
WDTCON	D	ADDR	00A7H	NOT USED
WDTPRG	D	ADDR	00A7H	NOT USED
WDRST	D	ADDR	00AGH	NOT USED
WR	B	ADDR	00B6H	NOT USED
<				

Ao observar o seu conteúdo é possível notar que há em sua construção, nomes dos símbolos, seus respectivos tipos e todos os demais atributos, como os endereçamentos de memória e seus respectivos valores.

Os caracteres presentes nas duas primeiras colunas representam respectivamente, o endereçamento de memória e o seu conteúdo.

- **Caracteres presentes na segunda coluna:**
 - **B:** Refere-se à memória “bit”.
 - **I:** Refere-se à memória “idata”.
 - **C:** Refere-se à memória “code”.
 - **D:** Refere-se à memória de “dados”.
 - **X:** Refere-se à memória “xdata”.

- **Abreviações presentes na terceira coluna:**
 - **ADDR:** Se refere a um endereço de memória.
 - **NUMB:** Se refere a um número.
 - **SEG:** Se refere a um segmento.

Nas duas últimas colunas é possível verificar o termo “NOT USED”.

4. RESULTADOS E DISCUSSÃO

Após todo o experimento foi possível compreender algumas peculiaridades da linguagem a nível de máquina “Assembly”. Após a execução do código listado foi possível compreender a relação entre as operações realizadas, o painel de LED 's e a atuação dos registradores entre si.

5. CONCLUSÕES

Foi possível concluir que a linguagem Assembly em atuação ao microcontrolador MCU 8051 pode dar uma vasta biblioteca de opções e atuações diante de seus registradores e portas, assim como demonstrado no decorrer do relatório com o exemplo do painel de LED 's.

6. REFERÊNCIAS BIBLIOGRÁFICAS

NETO, Hugo Vieira. MICROCONTROLADORES MCS51. Curitiba, 2002. Disponível em: <https://pessoal.dainf.ct.utfpr.edu.br/hvieir/download/mcs51.pdf>. Acesso em: 24 jul. 2021.

MORAIS, Misael. Organização e arquitetura de computadores. [S. l.], . 2021. Disponível em:

<https://drive.google.com/drive/folders/0Bwjlecok7TpyfnAtQmx6bE4yZ043amNsbnRxMkF4UFfpWVZhWmRfeVBSHRRVi1xRzNOZnM?resourcekey=0-WmQ6S1i6hLYyCoGBLbMeGw>. Acesso em: 01 ago. 2021