

Nome: Lucas da Silva Carrasco	R.A.: 22.120.053-8
Nome: William Yang	R.A.: 22.121.043-8

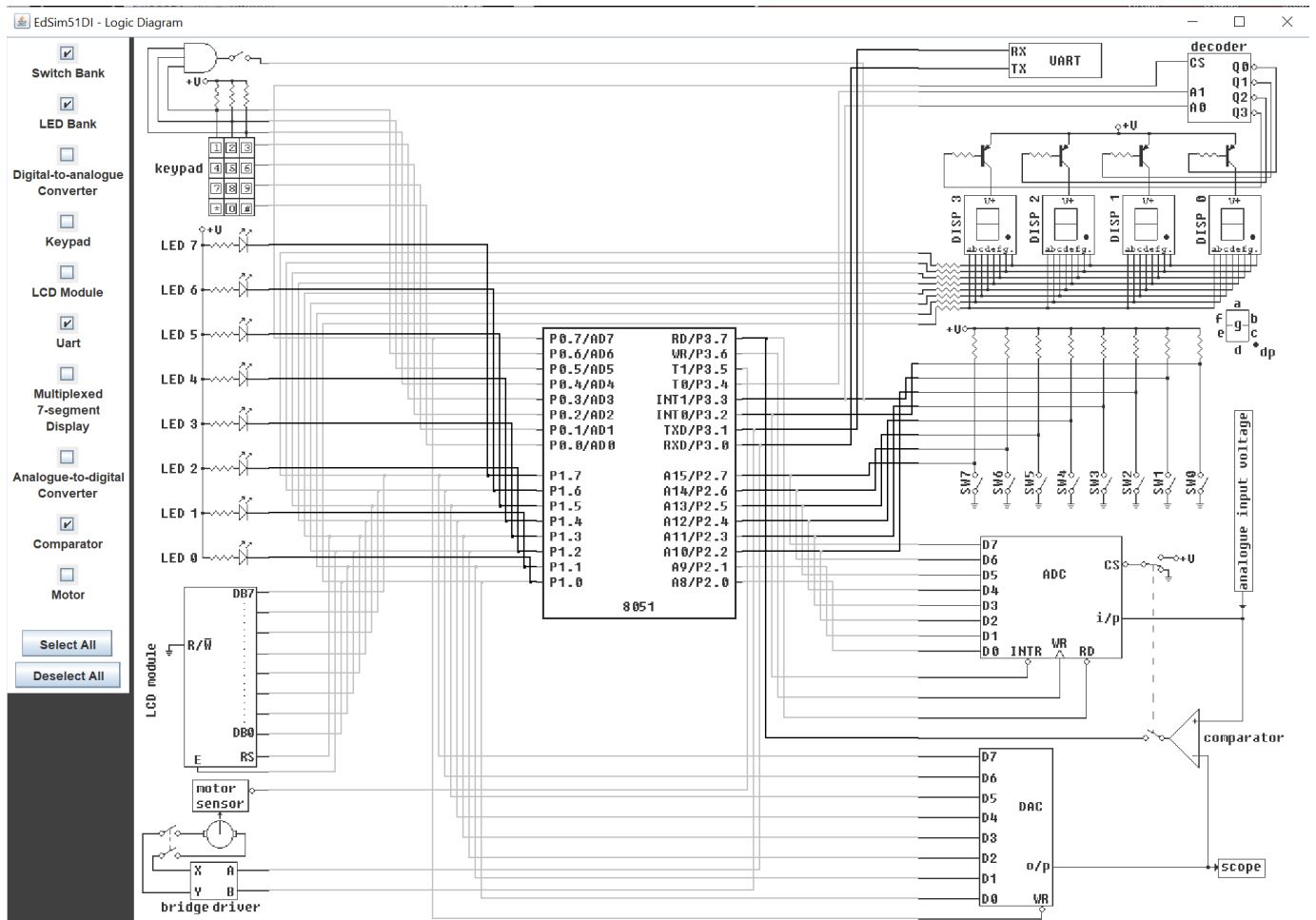
Projeto de Arquitetura de Computadores

1. Descrição do Projeto

O projeto realizado pelo grupo foi de uma fechadura eletrônico utilizando o microcontrolador 8051 simulado no EdSim51 Onde o usuário digitaria uma combinação alfanumérica de 4 dígitos na porta serial RS-232 e o programa ira escrever cada byte no endereço de memória, assim registrando a senha e “travando” a fechadura (indicado pelo pino p2.0 e p2.1 onde 1 = travado e 0 = aberto). Apos o registro da senha o usuário precisara inserir a combinação correta de 4 dígitos para abrir a fechadura, caso erro o pino p1.0 ira ficar piscando indicando que a senha esta incorreta e apos de um tempo quando parar de piscar o usuário vai poder tentar inserir a senha de novo. Quando a combinação inserida for a correta, os pinos p2.0 e p2.1 iram trocar seu estado para 0 indicando a abertura e todos os pinos p1 iram acender por um tempo e apagar.

2. Desenhos esquemáticos

O desenho esquemático do edsim51, mostrando as partes que estão sendo utilizadas.



3. Fluxograma ou Diagrama

Fluxograma ou desenho descrevendo as posições de memória, sub-rotinas, etc.;

System Clock (MHz)

SBUF

R/O	W/O	TH0	TL0
0x00	0x00	0x00	0x00
RXD	TXD	TMOD	0x20
1	1	TCON	0xC0
SCON	0x50		

pins bits

0xFF	0xFF	P3	0xF3	0xF6
0xFC	0xFC	P2		
0xFF	0xFF	P1		
0xFF	0xFF	P0		

TH1 TL1

0xF3	0xF6
------	------

PC

0x0223

8051

R7 0x06
R6 0x06
R5 0x06
R4 0xFF
R3 0x03
R2 0x05
R1 0x35
R0 0x30

B 0x00
ACC 0x00
PSW 0x00
IP 0x00
IE 0x90
PCON 0x80
DPH 0x00
DPL 0x00
SP 0x07

Modify RAM

Data Memory

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	35	05	03	FF	06	06	06	23	02	49	00	00	00	00	00
10	00	00	00	00	00	FF	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	61	61	61	61	61	0D	00	00	00	00	00	00	00	00	00	00
40	61	61	61	61	61	0D	00	00	00	00	00	00	00	00	00	00
50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2022 James Rogers

R0 ~ R7 = contadores para o programa

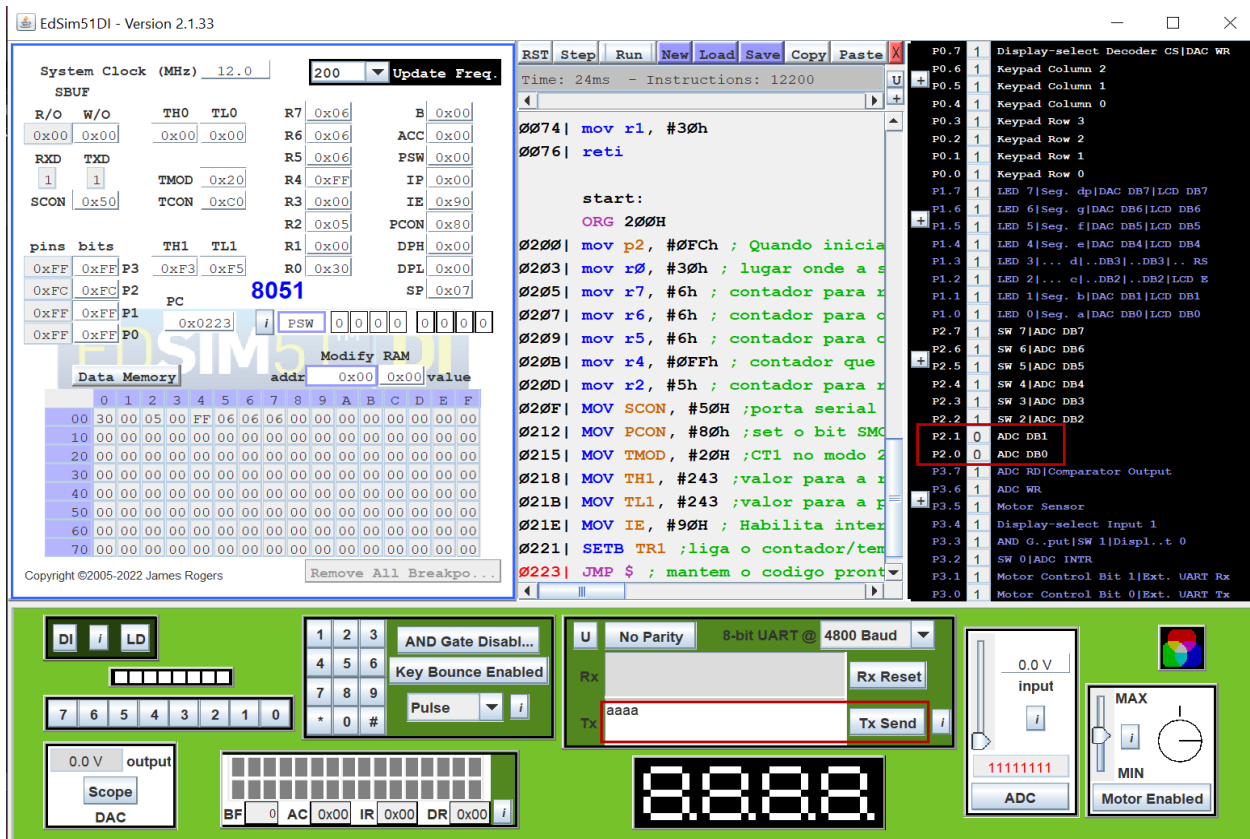
Endereço 30 ~ 33 da memória = registro da senha

Endereço 40~ 43 = tentativas

4. Imagens da simulação realizada na IDE

Algumas Imagens da simulação realizada na IDE, com telas apresentando os resultados obtidos.

Primeiro instante onde o programa esta esperando o registro da senha e com as portas p2.0 e p2.1 no estado 0 (Indicados pelos retângulos vermelho).



O programa registrando a senha nos endereço de memória de 30 – 33, e mudando o estado dos pinos p2.0 e p2.1 para 1.

Obs. A letra “a” minúscula equivale a 61 na tabela ASCII.

EdSim51DI - Version 2.1.33

System Clock (MHz) 12.0 200 Update Freq.

SBUF

R/O	W/O	TH0	TL0	R7	0x01	B	0xFF
0x0D	0x00	0x00	0x00	R6	0x06	ACC	0x0D
RXD	TXD	TMOD	0x20	R5	0x01	PSW	0x01
1	1	TCN	0xC0	R4	0xFF	IP	0x00
SCON	0x54	PCON	0x80	R3	0x0A	IE	0x90
				R2	0x05	DPH	0x00
				R1	0x05	DPL	0x00
				R0	0x35	SP	0x07

pins bits

0xFF	0xFF	P3	0xF3	0xFD
0xFF	0xFF	P2		
0xFF	0xFF	P1		
0xFF	0xFF	P0		

TH1 TL1

0x0223	0x0223
--------	--------

PC

8051

Modify RAM

addr	0x00	0x00	value
0	1	2	3
00	35	05	0A
01	06	01	23
02	00	00	00
03	00	00	00
04	00	00	00
05	00	00	00
06	00	00	00
07	00	00	00
08	00	00	00
09	00	00	00
0A	00	00	00
0B	00	00	00
0C	00	00	00
0D	00	00	00
0E	00	00	00
0F	00	00	00
10	00	00	00
11	00	00	00
12	00	00	00
13	00	00	00
14	00	00	00
15	00	00	00
16	00	00	00
17	00	00	00
18	00	00	00
19	00	00	00
1A	00	00	00
1B	00	00	00
1C	00	00	00
1D	00	00	00
1E	00	00	00
1F	00	00	00
20	00	00	00
21	00	00	00
22	00	00	00
23	00	00	00
24	00	00	00
25	00	00	00
26	00	00	00
27	00	00	00
28	00	00	00
29	00	00	00
2A	00	00	00
2B	00	00	00
2C	00	00	00
2D	00	00	00
2E	00	00	00
2F	00	00	00
30	61	61	61
31	61	61	61
32	0D	00	00
33	00	00	00
34	00	00	00
35	00	00	00
36	00	00	00
37	00	00	00
38	00	00	00
39	00	00	00
3A	00	00	00
3B	00	00	00
3C	00	00	00
3D	00	00	00
3E	00	00	00
3F	00	00	00

Time: 40ms 765us - Instructions: 20600

```

0074| mov r1, #30h
0076| reti

start:
ORG 200H
0200| mov p2, #0FCh ; Quando inicia
0203| mov r0, #30h ; lugar onde a s
0205| mov r7, #6h ; contador para r
0207| mov r6, #6h ; contador para c
0209| mov r5, #6h ; contador para c
020B| mov r4, #0FFh ; contador que
020D| mov r2, #5h ; contador para r
020F| MOV SCON, #50H ;porta serial
0212| MOV PCON, #80h ;set o bit SMC
0215| MOV TMOD, #20H ;CT1 no modo 2
0218| MOV TH1, #243 ;valor para a r
021B| MOV TL1, #243 ;valor para a p
021E| MOV IE, #90H ; Habilita inter
0221| SETB TR1 ;liga o contador/tem
0223| JMP $ ; mantem o codigo pront

```

P0.7	1	Display-select Decoder CS DAC WR
P0.6	1	Keypad Column 2
P0.5	1	Keypad Column 1
P0.4	1	Keypad Column 0
P0.3	1	Keypad Row 3
P0.2	1	Keypad Row 2
P0.1	1	Keypad Row 1
P0.0	1	Keypad Row 0
P1.7	1	LED 7 Seg. dp DAC DB7 LCD DB7
P1.6	1	LED 6 Seg. g DAC DB6 LCD DB6
P1.5	1	LED 5 Seg. f DAC DB5 LCD DB5
P1.4	1	LED 4 Seg. e DAC DB4 LCD DB4
P1.3	1	LED 3 ... d ..DB3 ..DB3 .. RS
P1.2	1	LED 2 ... c ..DB2 ..DB2 LCD E
P1.1	1	LED 1 Seg. b DAC DB1 LCD DB1
P1.0	1	LED 0 Seg. a DAC DB0 LCD DB0
P2.7	1	SW 7 ADC DB7
P2.6	1	SW 6 ADC DB6
P2.5	1	SW 5 ADC DB5
P2.4	1	SW 4 ADC DB4
P2.3	1	SW 3 ADC DB3
P2.2	1	SW 2 ADC DB2
P2.1	1	ADC DB1
P2.0	1	ADC DB0
P3.7	1	ADC RD Comparator Output
P3.6	1	ADC WR
P3.5	1	Motor Sensor
P3.4	1	Display-select Input 1
P3.3	1	AND G. put SW 1 Displ..t 0
P3.2	1	SW 0 ADC INTR
P3.1	1	Motor Control Bit 1 Ext. UART Rx
P3.0	1	Motor Control Bit 0 Ext. UART Tx

Copyright ©2005-2022 James Rogers

Remove All Breakpo...

DI i LD

7 6 5 4 3 2 1 0

0.0 V output

Scope

DAC

1 2 3

4 5 6

7 8 9

* 0 #

AND Gate Disabl...

Key Bounce Enabled

Pulse

U No Parity 8-bit UART @ 4800 Baud

Rx Rx Reset

Tx Tx Reset

0.0 V input

11111111

ADC

MAX

MIN

Motor Enabled

BF 0 AC 0x00 IR 0x00 DR 0x00

8888

Quando a senha, registrada no endereço de 40 – 43, for correta, os acendera os pinos p1 por um período e mudara o estado dos pinos p2.0 e p2.1 para 0 indicando a abertura .

EdSim51DI - Version 2.1.33

System Clock (MHz) 12.0 200 Update Freq.

SBUS

R/O	W/O	TH0	TL0	R7	B
0x0D	0x00	0x00	0x00	0xFB	0xD
RXD	TXD	TMOD	0x20	R6	ACC
1	1	0x20	0x00	0x06	0xFF
SCON	0x00	TCON	0xC0	R5	PSW
				0x01	0x00
				R4	IP
				0xD5	0x00
				R3	IE
				0x09	0x90
				R2	PCON
				0x00	0x80
				R1	DPH
				0x35	0x00
				R0	DPL
				0x45	0x00
					SP
					0x0B

pins bits TH1 TL1 R0 R1 R2 R3 R4 R5 R6 R7 B

0xFF 0xFF P3 0xFF 0xFF P2 8051 P1 0x00 P0 0xFF

Data Memory

addr	0x00	0x01	value
0	45	35	00
1	09	D5	01
2	06	FB	23
3	02	49	00
4	00	00	00
5	00	00	00
6	00	00	00
7	00	00	00
8	00	00	00
9	00	00	00
A	00	00	00
B	00	00	00
C	00	00	00
D	00	00	00
E	00	00	00
F	00	00	00

Modify RAM

Remove All Breakpo...

Assembly Code:

```

004A| cpl p1.0 ; indicador de led q
004C| MOV SCON, #0H ; desativa a po
004F| djnz r4, erro ; r4 e r3 funci
0051| djnz r3, erro
0053| jmp restart

acerto:
0055| mov p2, #0FCh ; abre a tranc
0058| mov p1, a ; serve para acende
005A| xch a, 15h ; a troca dos valc
005C| MOV SCON, #0H
005F| djnz r4, acerto
0061| djnz r3, acerto
0063| jmp restart

restart: ; depois de passar o
0065| mov p2, #0FFh ; trava a tranc
0068| mov r2, #5h ; todas as funcõe
006A| mov p1, #0FFh
006D| mov r6, #6h
  
```

IO Pins:

- P0.7: 1 Display-select Decoder CS|DAC WR
- P0.6: 1 Keypad Column 2
- P0.5: 1 Keypad Column 1
- P0.4: 1 Keypad Column 0
- P0.3: 1 Keypad Row 3
- P0.2: 1 Keypad Row 2
- P0.1: 1 Keypad Row 1
- P0.0: 1 Keypad Row 0
- P1.7: 0 LED 7|Seg. dp|DAC DB7|LCD DB7
- P1.6: 0 LED 6|Seg. g|DAC DB6|LCD DB6
- P1.5: 0 LED 5|Seg. f|DAC DB5|LCD DB5
- P1.4: 0 LED 4|Seg. e|DAC DB4|LCD DB4
- P1.3: 0 LED 3|... d|...DB3|...DB3|...RS
- P1.2: 0 LED 2|... c|...DB2|...DB2|...LCD E
- P1.1: 0 LED 1|Seg. b|DAC DB1|LCD DB1
- P1.0: 0 LED 0|Seg. a|DAC DB0|LCD DB0
- P2.7: 1 SW 7|ADC DB7
- P2.6: 1 SW 6|ADC DB6
- P2.5: 1 SW 5|ADC DB5
- P2.4: 1 SW 4|ADC DB4
- P2.3: 1 SW 3|ADC DB3
- P2.2: 1 SW 2|ADC DB2
- P2.1: 0 ADC DB1
- P2.0: 0 ADC DB0
- P3.7: 1 ADC RD|Comparator Output
- P3.6: 1 ADC WR
- P3.5: 1 Motor Sensor
- P3.4: 1 Display-select Input 1
- P3.3: 1 AND G..put|SW 1|Displ..t 0
- P3.2: 1 SW 0|ADC INTR
- P3.1: 1 Motor Control Bit 1|Ext. UART Rx
- P3.0: 1 Motor Control Bit 0|Ext. UART Tx

Hardware Interface:

- DI, LD, 7-segment display (0-9)
- AND Gate Disabl..., Key Bounce Enabled, Pulse
- 8-bit UART @ 4800 Baud, No Parity, Rx Reset, Tx Reset
- 0.0 V output, Scope, DAC
- Error! Function set not called.
- 0.0 V input, 11111111, ADC, Motor Enabled

Em caso de erro o programa ira piscar o pino p1.0 um tempo para depois poder receber o input de novo.

EdSim51DI - Version 2.1.33

System Clock (MHz) 12.0

200 Update Freq.

SBUF

R/O	W/O	TH0	TL0	R7	0x00	B	0x61	
0x62	0x00	0x00	0x00	R6	0x05	ACC	0x62	
RXD	TXD			R5	0x01	PSW	0x01	
1	1	TMOD	0x20	R4	0x5A	IP	0x00	
SCON	0x00	TCON	0x00	R3	0x03	IE	0x90	
				R2	0x05	PCON	0x80	
pins	bits	TH1	TL1	R1	0x31	DPH	0x00	
0xFF	0xFF	P3	0xF3	0xFA	R0	0x41	DPL	0x00
0xFF	0xFF	P2				SP	0x09	
0xFE	0xFE	P1						
0xFF	0xFF	P0						

PC 8051

Modify RAM

Data Memory

addr	0x00	0x00	value
0	41	31	05
1	03	05	01
2	05	00	23
3	02	00	00
4	00	00	00
5	00	00	00
6	00	00	00
7	00	00	00
8	00	00	00
9	00	00	00
A	00	00	00
B	00	00	00
C	00	00	00
D	00	00	00
E	00	00	00
F	00	00	00

Remove All Breakpo...

RST Step Run New Load Save Copy Paste

Time: 102ms 798us - Instructions: 52600

```

004A| cpl p1.0 ; indicador de led q
004C| MOV SCON, #0H ; desativa a p
004F| djnz r4, erro ; r4 e r3 funci
0051| djnz r3, erro
0053| jmp restart

acerto:
0055| mov p2, #0FCh ; abre a tranca
0058| mov p1, a ; serve para acende
005A| xch a, 15h ; a troca dos valc
005C| MOV SCON, #0H
005F| djnz r4, acerto
0061| djnz r3, acerto
0063| jmp restart

restart: ; depois de passar c
0065| mov p2, #0FFh ; trava a tranc
0068| mov r2, #5h ; todas as funcõe
006A| mov p1, #0FFh
006D| mov r6, #6h

```

P0.7	1	Display-select Decoder CS DAC WR
P0.6	1	Keypad Column 2
P0.5	1	Keypad Column 1
P0.4	1	Keypad Column 0
P0.3	1	Keypad Row 3
P0.2	1	Keypad Row 2
P0.1	1	Keypad Row 1
P0.0	1	Keypad Row 0
P1.7	1	LED 7 Seg. dp DAC DB7 LCD DB7
P1.6	1	LED 6 Seg. g DAC DB6 LCD DB6
P1.5	1	LED 5 Seg. f DAC DB5 LCD DB5
P1.4	1	LED 4 Seg. e DAC DB4 LCD DB4
P1.3	1	LED 3 ... d ...DB3 ...DB3 ... RS
P1.2	1	LED 2 ... c ...DB2 ...DB2 LCD E
P1.1	1	LED 1 Seg. b DAC DB1 LCD DB1
P1.0	0	LED 0 Seg. a DAC DB0 LCD DB0
P2.7	1	SW 7 ADC DB7
P2.6	1	SW 6 ADC DB6
P2.5	1	SW 5 ADC DB5
P2.4	1	SW 4 ADC DB4
P2.3	1	SW 3 ADC DB3
P2.2	1	SW 2 ADC DB2
P2.1	1	ADC DB1
P2.0	1	ADC DB0
P3.7	1	ADC RD Comparator Output
P3.6	1	ADC WR
P3.5	1	Motor Sensor
P3.4	1	Display-select Input 1
P3.3	1	AND G..put SW 1 Displ..t 0
P3.2	1	SW 0 ADC INTR
P3.1	1	Motor Control Bit 1 Ext. UART Rx
P3.0	1	Motor Control Bit 0 Ext. UART Tx

DI i LD

7 6 5 4 3 2 1 0

0.0 V output

Scope

DAC

1 2 3

4 5 6

7 8 9

* 0 #

AND Gate Disabl...

Key Bounce Enabled

Pulse

U No Parity 8-bit UART @ 4800 Baud

Rx Rx Reset

Tx Tx Reset

0.0 V input

1111111

ADC

MAX

MIN

Motor Enabled

BF 0 AC 0x00 IR 0x00 DR 0x00

8.8.8.8

5. Discussões e conclusões

Encontramos várias dificuldades durante o projeto durante a implementação da lógica, por exemplo, na parte de como realizar o registro da primeira leitura e pular para outra linha para registrar as tentativas e sobrescrever elas em caso de erro, realizar a rotina para ele somente só fazer a verifica-o das tentativas sem afetar a senha registrada e muitos mais.

Percebemos que a programação em linguagem assembly realmente é uma coisa diferente, precisando de resolver coisas de um jeito que não funciona em outra linguagem.

6. Código-fonte

```
org 000h
jmp start
org 023H ; PONTEIRO DA INTERRUPCAO PARA CANAL SERIAL
mov p2, #0FFh ; quando entrar na interrupção a tranca é fechada
djnz r7,escrita ; registro da senha
mov r1, #30h ; lugar que vai registrar a senha
mov r0, #40h ; indicador usado para comparação de senhas
djnz r6, escrita ; confirmacao entra aqui para ele entrar de novo na func de escrever
reti

escrita: ; funciona tanto para registro de senha quanto para a comparação
mov r3, #0Ah ; um timer para os acertos e erros
MOV A,SBUF ; REALIZA A LEITURA DO BYTE RECEBIDO
MOV @r0, A ; ESCRIVE O VALOR NO ENDEREÇO 30H
mov a, @r0 ; na primeira passagem r0 registra a senha, na segunda é usada para a
comparação também
mov b, @r1 ; r1 utilizado para a comparação de senhas
inc r0 ; os inc's fazem parte tanto do registro quanto da comparação
inc r1
CLR RI ; RESETA RI PARA RECEBER NOVO BYTE
djnz r5, return ; utilizado para fazer o registro e travar o código, para o mesmo não chegar na
comparação no primeiro input
inc r5 ; mantém o r5 em 1 para não quebrar o código.
cjne a, b, erro ; se a senha digitada for diferente da senha registrada cjne chama a subrotina
erro
djnz r2, return ; r2 conta os acertos até o fim da senha
mov 15h, #0ffh ; serve para o led do indicador de acertos
clr a
acall acerto

return:
reti
erro:
cpl p1.0 ; indicador de led que mostra o erro, piscando apenas o primeiro Led
MOV SCON, #0H ; desativa a porta serial para não receber nada até o fim do tempo de espera
djnz r4, erro ; r4 e r3 funcionam como timers
djnz r3, erro
jmp restart

acerto:
mov p2, #0FCh ; abre a tranca, localizada no P2.0 e P2.1
mov p1, a ; serve para acender/apagar todos os Leds
xch a, 15h ; a troca dos valores dentro da função faz os leds piscarem
MOV SCON, #0H
djnz r4, acerto
```

```
djnz r3, acerto  
jmp restart
```

restart: ; depois de passar do acerto ou erro, o restart prepara o programa para receber outra tentativa de acesso

```
mov p2, #0FFh ; trava a tranca caso tenha passado pelo acerto  
mov r2, #5h ; todas as funções até o reti resetam o programa para receber outra senha  
mov p1, #0FFh  
mov r6, #6h  
MOV SCON, #50H  
mov r0, #40h  
mov r1, #30h  
reti
```

start:

ORG 200H

```
mov p2, #0FCh ; Quando iniciado a tranca fica aberta até o registro da senha  
mov r0, #30h ; lugar onde a senha vai ser salvo  
mov r7, #6h ; contador para registrar os 4 numeros  
mov r6, #6h ; contador para os input  
mov r5, #6h ; contador para conferir a senha  
mov r4, #0FFh ; contador que funciona como um "Timer"  
mov r2, #5h ; contador para registrar os acertos  
MOV SCON, #50H ;porta serial no modo 1 e habilita a recepção  
MOV PCON, #80h ;set o bit SMOD  
MOV TMOD, #20H ;CT1 no modo 2  
MOV TH1, #243 ;valor para a recarga  
MOV TL1, #243 ;valor para a primeira contagem  
MOV IE, #90H ; Habilita interrupção serial  
SETB TR1 ;liga o contador/temporizador 1  
JMP $ ; mantem o codigo pronto para o input
```