Ramon Elias do Patrocínio Raymundo – 2020015757

1. DESCRIÇÃO DO PROJETO

O projeto em questão trata-se de um sistema embarcado que é capaz de controlar a temperatura(resfriamento) e o nível de umidade do solo de uma estufa para plantas, que, mais especificamente, é um cultivo de Heléboros.

Contudo, foi necessário a utilização de Conversores A/D para lermos a variação dos sensores, que no caso foram substuídos por potenciômetros para fins de simulação no PicSimLab, Display LCD para mostrar os níveis de tensão gerados pelos sensores de umidade e temperatura e a temperatura em graus Celsius, Leds para a simular o acionamento de três bombas e a Ventoinha que será acionada com diferentes rotações de acordo com o aumento da temperatura, para as diferentes rotações utilizamos uma saída PWM.

Figura 1:Ilustração para sensores de temperatura, umidade, potenciômetro,ventoinha e microcontrolador PIC18F4520.

2. DETALHAMENTO DO LCD, PIC184520 E SENSORES

O Display LCD 16x2 disponível no PicSimlab tem como chip controlador o HD44780, chip desenvolvido pela Hitachi. Por ser um chip de qualidade, acabou se tornando padrão para vários modelos. A comunicação entre o Display e o microcontrolador(PIC18F4520) é feita através da interface I2C.

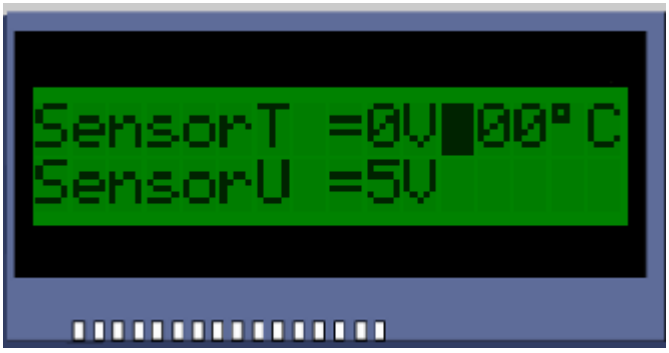

Figura 2: Display LCD do PICSimLab

O microcontrolador PIC18f4520 faz parte da família de microcontroladores de 8 bits e núcleo de 14 bits (série PIC18F...) fabricada pela MICROCHIP. Este PIC contém 256 bytes de memória EEPROM e 32Kb de memória Flash.

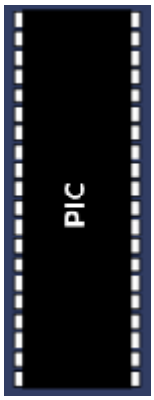

Figura 3: PIC do PICSimLab

O sensor de umidade funciona do seguinte modo, quando a terra/solo está seca, ela possuí alta resistência, dificultando a condução de corrente elétrica. Quando adicionamos água, a mesma, por sua vez, diminuí essa resistência e conduz com mais facilidade corrente elétrica. Portanto no projeto, 0 V proviniente do sensor de umidade equivale a um solo seco e 5 V um solo úmido. Do mesmo modo que o sensor anterior, o sensor de temperatura também gerava uma tensão de 0 a 5 V, 0 V equivale a 0°C e 5 V equivale 50°C, tendo, portanto, uma variação de 0,1 V/°C. Para simular estes sensores foi utilizado os potenciômetros disponibilizados no PicSimLab.

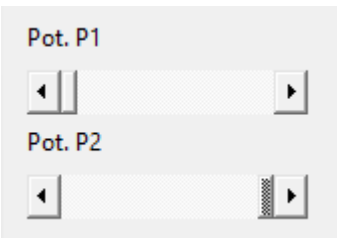

Figura 4: Potenciômetros do PICSimLab

3. DIFICULDADES

No desenvolvimento do projeto uma dificuldade foi encontrada, a biblioteca ADC(conversor analógico-digital) utilizada apresentava um erro no seu diretório .c, impossibilitando a leitura do canal 2(Pot.P2 do PicSimLab). Segue abaixo uma figura com sua resolução.

```
unsigned int adc_amostra(unsigned char canal) {

    switch (canal) {
        case 0:
            ADCON0 = 0x01;
            break;
        case 1:
            ADCON0 = 0x09;
            break;
        case 2:
            ADCON0 = 0x05;
            break;
    }
```

Figura 5: Código

4. DESENVOLVIMENTO

Primeiramente, foram adicionadas as bibliotecas que o código iria necessitar, são elas:

```
1  #include "pic18f4520.h"
2  #include "config.h"
3  #include "atraso.h"
4  #include "lcd.h"
5  #include "adc.h"
6  #include "i2c.h"
7  #include "pwm.h"
8  #include "bits.h"
```

Figura 6: Bibliotecas

Logo após foram executadas duas funções para converter os valores provinientes dos potenciômetros (simulação dos sensores) , em seguida abriu-se a função main( ), na qual foram criadas as variáveis , configuração dos TRIS(es), inicialização do LCD, ADC e PWM.

```
10    void itoa(unsigned int val, char* str );
11    void itoa1(unsigned int val1, char* str1 );
12
13    void main() {
14
15        unsigned char tmp, umid,temp;
16        unsigned char tensao;
17        char str[6],grau = 223;
18        char str1[6];
19
20        ADCON1 = 0x06;
21        TRISA = 0xC3;
22        TRISB = 0xF0;
23        TRISC = 0x00;
24        TRISD = 0x00;
25        TRISE = 0x00;
26
27        lcd_init();
28        adc_init();
29        pwmInit();
```

Figura 7: Código

Após todas as configurações e inicializações feitas, temos a excecução em si do código dentro de um loop infinito, onde a ventoinha será acionada pela saída PWM de acordo com o aumento da temperatura e mostrará informações no LCD, acionamento de 3 leds do PORTB,b1,b2,b3 simulando três bombas, todas ligadas com solo seco, duas ligadas com o solo com pouca presença de água, uma ligada com o solo relativamente úmido e todas desligadas quando o solo estiver úmido. Para o funcionamento de todas essas funções, devemos apertar o botão RB5. Segue abaixo o restante do código que controla estas saídas.

```
32    lcd_cmd(L_L1);
33    lcd_str("Ligue o circuito");
34    lcd_cmd(L_L2);
35    lcd_str("pressione RB5");
36    while (PORTBbits.RB5);
37    lcd_cmd(L_CLR);
38    lcd_cmd(L_L1);
39    lcd_str("SensorT = ");
40    lcd_cmd(L_L2);
41    lcd_str("SensorU = ");
42
43    for (;;) {
44        umid = (adc_amostra(2)*10) / 204;
45        tmp = (adc_amostra(0)*10) / 204;
46        lcd_cmd(L_L1 + 9 );
47        itoa(tmp, str);
48        itoa1(umid, str1);
49        tensao = str[3];
50        umid = str1[3];
51
52            if(tensao <= 50 && tensao >= 49) pwmSet1(32);
53            if(tensao <= 52 && tensao > 50)  pwmSet1(64);
54            if(tensao <= 53 && tensao > 52)  pwmSet1(92);
55            if(tensao == 48) pwmSet1(0);
56            if(umid <= 50 && umid >= 49)
57            {
58               BitSet(PORTB,2);
59               BitSet(PORTB,3);
60               BitClr(PORTB,1);
61            }
```

```
62            if(umid <= 52 && umid > 50)
63            {
64               BitClr(PORTB,2);
65               BitSet(PORTB,3);
66               BitClr(PORTB,1);
67            }
68            if(umid <= 53 && umid > 52)
69            {
70               BitClr(PORTB,1);
71               BitClr(PORTB,3);
72               BitClr(PORTB,2);
73            }
74            if(umid == 48)
75            {
76                BitSet(PORTB,3);
77                BitSet(PORTB,2);
78                BitSet(PORTB,1);
79            }
80        lcd_dat(str[3]);
81        lcd_dat('V');
82        lcd_dat(' ');
83        lcd_dat(str[3]);
84        lcd_dat(str[4]);
85        lcd_dat(grau);
86        lcd_dat('C');
87        lcd_cmd(L_L2 + 9 );
88        lcd_dat(str1[3]);
89        lcd_dat('V');
90        atraso_ms(10);
91    }
```

```
92  }
93  void itoa(unsigned int val, char* str )
94  {
95    str[0]=(val/10000)+0x30;
96    str[1]=((val%10000)/1000)+0x30;
97    str[2]=((val%1000)/100)+0x30;
98    str[3]=((val%100)/10)+0x30;
99    str[4]=(val%10)+0x30;
100   str[5]=0;
101 }
102 void itoa1(unsigned int val1, char* str1 )
103 {
104   str1[0]=(val1/10000)+0x30;
105   str1[1]=((val1%10000)/1000)+0x30;
106   str1[2]=((val1%1000)/100)+0x30;
107   str1[3]=((val1%100)/10)+0x30;
108   str1[4]=(val1%10)+0x30;
109   str1[5]=0;
110 }
```

Figuras 8, 9 e 10: Restante do Código

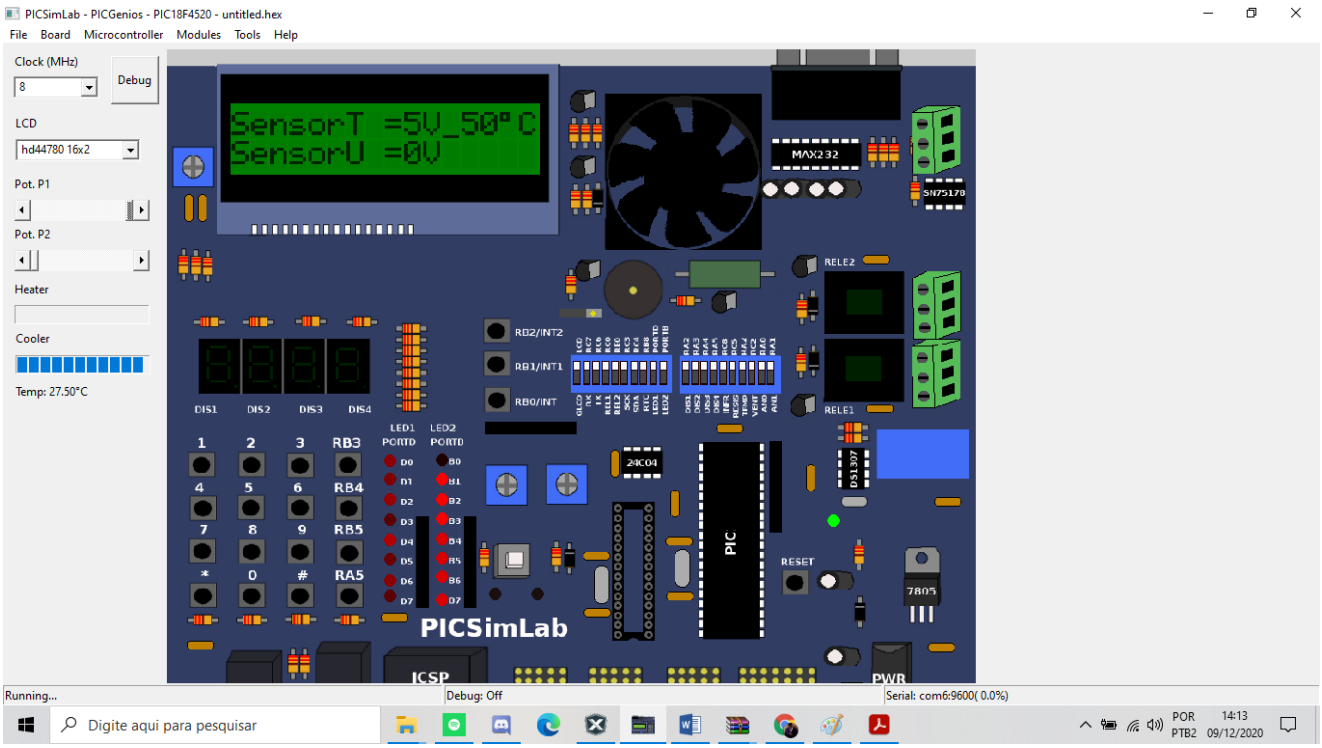

Figura 11: Simulação no PicSimLab

5. CONCLUSÃO

Podemos concluir que o projeto desenvolvido está de acordo com a proposta, já que utilizando o PICSimLab para simulação, executa o funcionamento de controle de umidade e temperatura de uma estufa para plantas e utiliza conceitos aprendidos na disciplina. Portanto, o projeto foi concluído com sucesso.