Lucas Ávila Silva – 35132

1. PROJETO

Este projeto tem o intuito de simular um sistema embarcado voltado para controlar uma estufa de plantas, alterando a temperatura e a umidade do ambiente em questão.

Para que isso fosse possível, foram utilizados: um display LCD, para mostrar a leitura dos sensores; conversores A/D para realizar a leitura dos sensores (que foram substituídos por potenciômetros para fim de simulação no software PicSimLab); a Ventoinha, utilizada para resfriar o ambiente; LEDs para representar o acionamento das bombas d'água, responsáveis por manter a umidade dentro do desejado.

Figura 1 - Ilustração dos componentes

2. SOBRE O PIC18F4520

O microcontrolador PIC18f4520 faz parte da família de microcontroladores de 8 bits com núcleo de 14 bits, fabricada pela Microchip. O mesmo possuí 256 bytes de memória EEPROM e 32kb de memória Flash.

A seguir é apresentado o PIC18f4520:

Figura 2 - Microcontrolador PIC18f4520

3. DISPLAY LCD E SENSORES

O display LCD utilizado é o LCD 16x2, disponível no PicSimLab. Este display possui o chip controlador HD44780, feito pela Hitachi. Este chip está presente em diversos modelos de LCD. O microcontrolador PIC18f4520 se comunica com o display utilizando a interface I2C.

A seguir é apresentado o display no simulador PicSimLab:

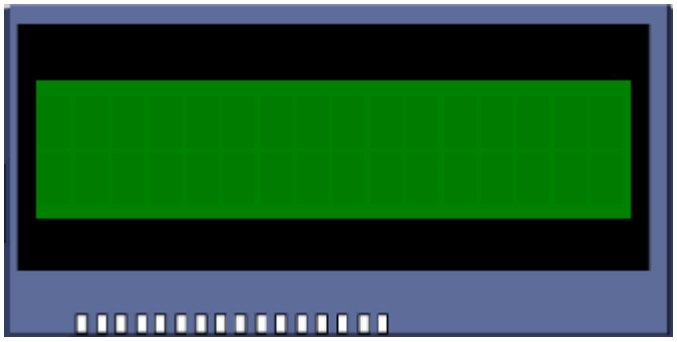

Figura 3 - Display LCD 16x2 PicSimLab

Os sensores utilizados são de Temperatura e Umidade e ambos variam de 0V até 5V. O sensor de Temperatura gera uma tensão de 0,1V/ºC, ou seja, quando a temperatura ambiente está em 0ºC, a leitura do sensor será de 0V, quando estiver em 10ºC, a leitura será de 1V e assim por diante. O sensor de umidade atua de acordo com a resistência da terra à condução de corrente, sendo assim, quanto mais seco, mais resistência e menor a tensão. Portanto 0V corresponde a um solo seco e 5V um solo úmido.

A seguir são mostrados os sensores, que foram substituídos pelos potenciômetros do PicSimLab.

Figura 4 - Potenciômetros PicSimLab

4. PRINCIPAL DIFICULDADE

Em um determinado momento, mesmo com o código supostamente certo, ainda era constatado um erro com relação ao canal de comunicação do Pot. P2. Após algumas pesquisas para descobrir qual era o problema, foi constatado que havia um erro na função **adc_amostra** biblioteca **adc.c**, responsável por fazer a conversão analógico-digital. O problema estava no **case 2**, que determinava o valor **0x01** para ADCON0. A correção foi feita substituindo o valor de **0x01** para **0x05**. Segue uma imagem com a 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 após a mudança

5. DESENVOLVIMENTO DO PROJETO

Primeiramente foram adicionadas todas a bibliotecas que continham as funções necessárias para desenvolver o código:

```
#include "picl8f4520.h"
#include "config.h"
#include "atraso.h"
#include "lcd.h"
#include "adc.h"
#include "i2c.h"
#include "pwm.h"
#include "bits.h"
```

Figura 6 - Bibliotecas

Em seguida foram declarados os protótipos das funções usadas para a conversão dos valores dos potenciômetros, foram feitas as designações dos valores para os TRIS e a inicialização de LCD, ADC e PWM.

```
void itoa(unsigned int val, char* str );
void itoa1(unsigned int val1, char* str1 );

void main() {

    unsigned char tmp, umid,temp;
    unsigned char tensao;
    char str[6],grau = 223;
    char str1[6];

    ADCON1 = 0x06;
    TRISA = 0xC3;
    TRISB = 0xF0;
    TRISC = 0x00;
    TRISD = 0x00;
    TRISE = 0x00;

    lcd_init();
    adc_init();
    pwmInit();
```

Figura 7 - Declarações iniciais

Então a lógica começou a ser implementada para o acionamento da ventoinha pela saída PWM de acordo com o valor do Pot. P1, sendo este valor mostrado no LCD. Posteriormente, foi implementado o acionamento dos 3 LEDs da PORTB (b1, b2, b3) de acordo com a leitura do Pot. P2, que simulam as bombas d'água. Por último, para determinar uma forma de inicialização do processo, foi definido o botão RB5 como "botão de ligar". O código é apresentado a seguir:

```
lcd_cmd(L_CLR);
lcd_cmd(L_L1);
lcd_str("Ligar o circuito");
lcd_cmd(L_L2);
lcd_str("pressionando RB5");
while (PORTBbits.RB5);
lcd_cmd(L_CLR);
lcd_cmd(L_L1);
lcd_str("Sensor1 = ");
lcd_cmd(L_L2);
lcd_str("Sensor2 = ");

for (;;) {
    umid = (adc_amostra(2)*10) / 204;
    tmp = (adc_amostra(0)*10) / 204;
    lcd_cmd(L_L1 + 9 );
    itoa(tmp, str);
    itoa1(umid, str1);
    tensao = str[3];
    umid = str1[3];

        if(tensao <= 50 && tensao >= 49) pwmSet1(32);
        if(tensao <= 52 && tensao > 50)  pwmSet1(64);
        if(tensao <= 53 && tensao > 52)  pwmSet1(92);
        if(tensao == 48) pwmSet1(0);
        if(umid <= 50 && umid >= 49)
        {
           BitSet(PORTB,2);

            BitSet(PORTB,3);
            BitClr(PORTB,1);
        }
        if(umid <= 52 && umid > 50)
        {
            BitClr(PORTB,2);
            BitSet(PORTB,3);
            BitClr(PORTB,1);
        }
        if(umid <= 53 && umid > 52)
        {
            BitClr(PORTB,1);
            BitClr(PORTB,3);
            BitClr(PORTB,2);
        }
        if(umid == 48)
        {
            BitSet(PORTB,3);
            BitSet(PORTB,2);
            BitSet(PORTB,1);
        }
    lcd_dat(str[3]);
    lcd_dat('V');
    lcd_dat(' ');
    lcd_dat(str[3]);
    lcd_dat(str[4]);
    lcd_dat(grau);
    lcd_dat('C');
```

```
        lcd_cmd(L_L2 + 9 );
        lcd_dat(str1[3]);
        lcd_dat('V');
        atraso_ms(10);
    }
}
void itoa(unsigned int val, char* str )
{
  str[0]=(val/10000)+0x30;
  str[1]=((val%10000)/1000)+0x30;
  str[2]=((val%1000)/100)+0x30;
  str[3]=((val%100)/10)+0x30;
  str[4]=(val%10)+0x30;
  str[5]=0;
}
void itoa1(unsigned int vall, char* str1 )
{
  str1[0]=(vall/10000)+0x30;
  str1[1]=((vall%10000)/1000)+0x30;
  str1[2]=((vall%1000)/100)+0x30;
  str1[3]=((vall%100)/10)+0x30;
  str1[4]=(vall%10)+0x30;
  str1[5]=0;
}
```

Figura 8,9 e 10 - Código

Por fim, o projeto é simulado no PicSimLab:

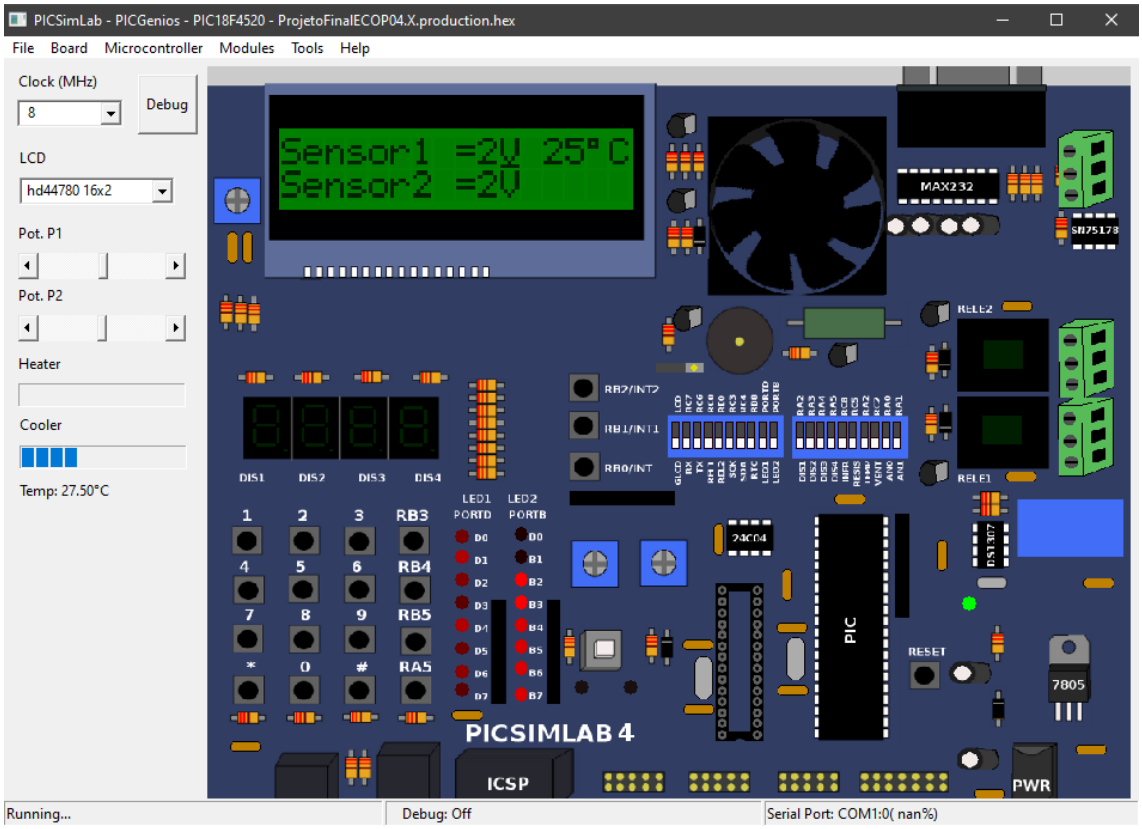

Figura 11 - Simulação

6. Conclusão

Após o desenvolvimento, pode-se concluir que o projeto desenvolvido atende 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.