Controle de temperatura PID
Malha fechada

Vivian Emanoelle Marinho Gomes
Igor da Silva Zagonel

IFMT – MT

igorzagonel@gmail.com
vivianemanoellegomes@gmail.com

Resumo **— Este relatório tem como objetivo familiarizar os alunos com as montagens e programações dos microcontroladores nas realizações de tarefas**.

I. Introdução

O objetivo do projeto é criar um controle PID para controlar a temperatura com o PIC 16F877A e plotar os sinais em um gráfico construído no processing. Para realizar esse projeto utilizamos apenas o PICSimlab e o processing.

II. A Teoria

- Controle PID

O controle proporcional é utilizado para minimizar a características de oscilação do controle de ligar / desligar. Ele vai além de reduzir erros e fornecer precisão e estabilidade no processo. Todos os três termos PID precisam ser ajustados corretamente aos requisitos da aplicação para assim conseguir um melhor controle. São controladores muito comum utilizados que na maioria dos casos demanda de um sistema proporcional, integrativo e derivativo, podem controlar algumas coisas como controle de temperatura, fluxo, vazão, pressão e etc. Normalmente possui uma entrada e uma saída.

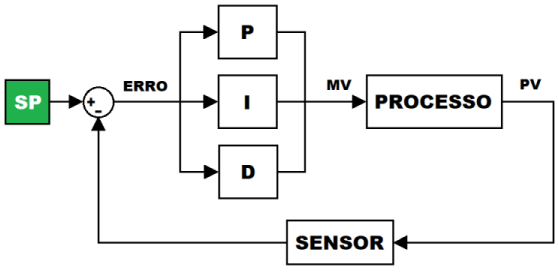

Fig. 1 – Controle PID

- PIC 16F877A

O PIC 16F877A é um microcontrolador da família de 8 bits, ou seja, lê apenas palavras de no máximo 8 bits e possui núcleo de 14 bits. A letra F indica que a memória programada desse PIC é do tipo Flash, cada linha da memória é uma palavra de 14 bits. Os três números finais do PIC permitem identifica-lo precisamente. A frequência de operação (clock) vai até 20MHz, memória RAM com 368 bytes e a memória EEPROM com 256 bytes, pode funcionar em uma alimentação de 2V a 5,5V e sua pinagem possui 40 pinos.

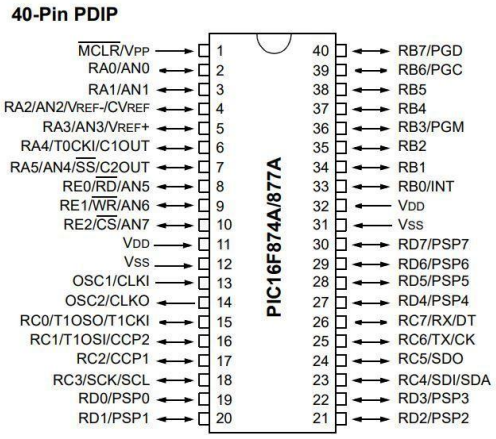

Fig. 2 – PIC 16F877A

- Processing

O processing é uma linguagem de programação de código aberto e ambiente de desenvolvimento integrado (IDE). É considerado um sketchbook, uma alternativa de projetos sem utilizar uma IDE padrão. Essa ferramenta possui grande facilidade na criação e manipulação de desenhos.

Fig. 3 – Processing.

III. Programação e Circuito

A montagem realizada nesse projeto consiste em um microcontrolador, o objetivo do mesmo era receber dois sinais diferentes enviados por potenciômetros e fazer a

plotagem no gráfico.

- Programas
  - Códigos PCW

A primeira parte está as criação das variaveis que foram criadas para a execução do projeto.

```
float TempRef, TempRes;
int16 pontDist;

char grau = 223;

float TempAnterior = 0;
float erro;
float kp = 10;
float ki = 0.2;
float kd = 0.005;
float proporcional;
float integrativo = 0;
float derivativo;
float PID;
```

Fig. 4 – Variáveis criadas

Nessa função faz a leitura analógica que vai de 0 a 1023 e converte para graus que nesse caso a referência é de 20ºC a 70ºC.

```
float InserirTempRef(){

   float TempPot;
   set_adc_channel(0);
   delay_us(50);
   TempPot = read_adc() * 0.0489 + 20;
   return TempPot;

}
```

Fig. 5 – Função temperatura de referência.

A função inserir distúrbio faz a leitura e devolve no pwm.

```
void InserirDisturbio(){

   set_adc_channel(1);
   delay_us(50);

   pontDist = read_adc();
   set_pwm1_duty(pontDist);
   delay_us(50);
}
```

Fig. 6 – Função inserir distúrbio.

A função temperatura da resistência transforma mili volt em graus o valor vai de 20 a 70.

```
float TempResistor(){
   float Temp;

   set_adc_channel(3);
   delay_us(50);
   Temp = read_adc() * 0.489;

   return Temp;

}
```

Fig. 7 – Função de temperatura da resistência.

A função de controle PID, calculando por tentativas e erros. E aqui estão os cálculos.

```
void ControlePID(){

   erro = TempRef - TempRes;

   proporcional = erro * kp;

   integrativo += erro * ki;

   derivativo = (TempAnterior - TempRes) * kd;

   TempAnterior = TempRes;

   PID = proporcional + derivativo + integrativo;

   if(PID < 1){
      PID = 0;
      integrativo = 0;
     set_pwm1_duty(512);
   }else if(PID > 1023){
      PID = 1023;
      set_pwm2_duty(PID);
   }else{
     set_pwm2_duty((int16)PID);
   }
}
```

Fig. 8 – Função controle PID.

Na função main está chamando as funções, escrevendo no lcd e mandando para o serial no processing.

```
while(TRUE){

   InserirDisturbio();
   TempRef = InserirTempRef();
   TempRes = TempResistor();
   ControlePID();
   printf(lcd_escreve, "\fTEMP: %f %cC\n", TempRes,grau);
   printf(lcd_escreve, "REF : %f %cC", TempRef,grau);

   fprintf(Wireless, "%f %f %f %f %Ld\n",TempRef,TempRes,PID,erro,pontDist);

   delay_ms(1000);
```

Fig. 9 – Função Main

IV. Comentários e conclusões

Com esse projeto concluímos que é possível construir um controle PID utilizando o PIC em uma linguagem simples. Ainda foi realizado uma plotagem no gráfico na qual mostrava o erro, temperatura de referência, temperatura do resistor que varia e o cooler. Fazendo também a comunicação serial e enviando para plotagem no gráfico.

V. REFERÊNCIAS

[1] West, “Controle PID”, Disponível em < https://www.west-cs.com.br/blog/o-que-e-controle-pid/>

[2]Microcontrolador, “Microcontrolador PID”, Disponível em < https://pt.wikipedia.org/wiki/Microcontrolador_PIC>

[2] Seu curso, “Processing”, Disponível em < http://www.seucurso.com.br/index.php?option=com_content&view=article&id=131:processing-o-que-e-e-para-que-serve&catid=32&Itemid=150>