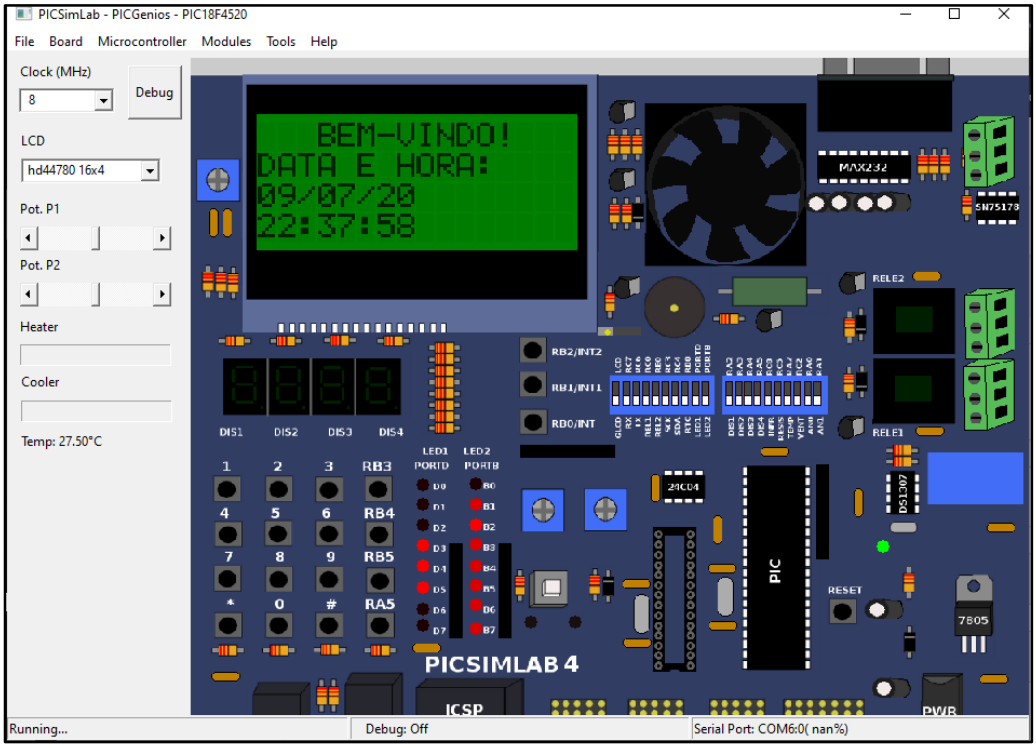
Autor(a): Stéfany Coura Coimbra / **Matrícula:** 2019008562

Curso: Engenharia de Computação, Unifei, Itajubá / **Disciplina:** ECOP14

Projeto de um Controlador de Temperatura PID utilizando MPLABX IDE e PICsimLab



Muitas indústrias utilizam o controlador de temperatura PID de modo a regular essa grandeza de um determinado processo, por exemplo, de uma caldeira em uma indústria metalúrgica. Há ainda um processo próximo de nós que pode ser observado todos os dias de controle de temperatura, que seria o de regulagem da temperatura da água de um chuveiro. Visando a importância do sistema, este foi modelado e implementado na disciplina de ECOP14 (Laboratório de Programação Embarcada), simulando um controlador de temperatura PID com o uso das ferramentas MPLABX IDE e PICSimLab, com o uso do microcontrolador PIC18F4520.

A proposta do projeto está relacionada à regulagem de temperatura utilizando o algoritmo de PID (Proporcional-Integral-Derivativo), com a variação de Kp, Ki e Kd a fim de ser obtida a resposta ideal da temperatura desejada (SetPoint).

Descrição

O valor desejado da temperatura (que idealmente em um sistema real seria medido por um sensor de temperatura especificado) no sistema (SetPoint) será definido pelo usuário utilizando as teclas do teclado matricial. Quatro botões serão dedicados ao menu de configuração das constantes Kp, Ki, Kd (para o controle PID), SetPoint, incremento e decremento do valor da temperatura medido manipulado, simulando um potenciômetro. Tal valor será apresentado para o usuário de tal forma a poder ser monitorado todo o sistema.

Quando o valor medido for diferente daquele desejado, de acordo com o propósito da grandeza para o processo ao qual o sistema está inserido, duas podem ser as situações: caso a temperatura real seja maior que a ideal, o cooler da placa e um LED de alerta serão ligados de maneira a abaixar esse

número; caso a temperatura real seja menor que a ideal, outro LED é acionado e também um relé (no processo em si em uma indústria, por exemplo, seria conectado a um aquecedor para aumentar o valor da temperatura). Um som será emitido com o buzzer caso a temperatura medida chegue ao SetPoint, juntamente a um LED, que seria para informar que a condição definida inicialmente foi satisfeita. O controle PID neste projeto é baseado no sistema de malha fechada, ou seja, o controle da variável observada é feito durante o processo. Esse controle faz com que "o processo de leitura do sensor para fornecer *feedback* constante e o cálculo para definir a saída desejada do atuador se repete continuamente a uma taxa fixa" (https://www.ni.com/pt-br/innovations/white-papers/06/pid-theory-explained.html).

Ferramentas e componentes utilizados no projeto

- Ambiente de programação para sistemas embarcados: MPLAB (link para download: https://www.microchip.com/mplab/mplab-x-ide)
- Compilador (link para download: https://www.microchip.com/en-us/developmenttools-tools-and-software/mplab-xc-compilers)
- Simulador PICSimLab (link para download: https://sourceforge.net/projects/picsim/)

  - PICSimLab, com PICGenios PIC18F4520
- Placa PICGenios com PIC18F4520
- LCD 16X4
- Teclado e botões RB3, RB4, RB5 E RA5
- LEDs
- Cooler
- Heater
- Relé
- Buzzer
- Display de sete segmentos

Passo 1: O Software

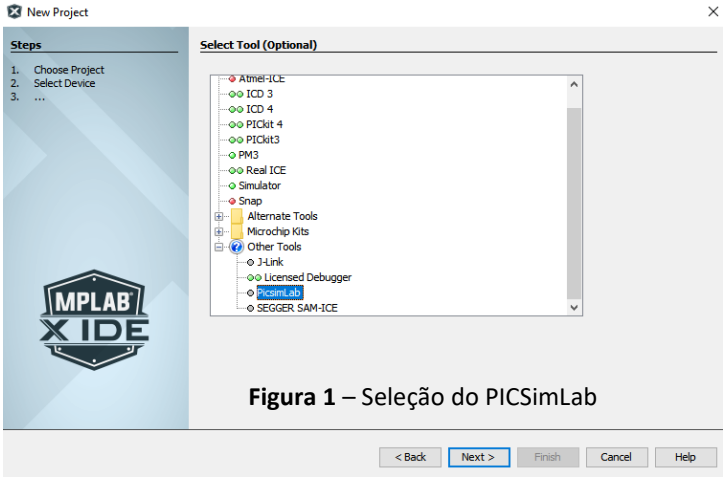


Figura 1 – Seleção do PICSimLab

Após a instalação dos softwares necessários, inicia-se a construção do projeto para que possa ser feito o código do projeto, em C. Para isso, inicialmente é criado um novo projeto no MPLAB X IDE, em *File* (ou Arquivo). Em *Categories*, é selecionado *Microship Embbeded*. Em *Projects*, *Standolone Project*. Na próxima janela, é escolhido o microcontrolador desejado, que para este projeto é o PIC18F4520. Na janela *Select Tool*, selecione o PicSimLab. Para que apareça como uma opção, foi anteriormente configurado em *Tools* → *Plugins*. É selecionado o compilador XC8 e salvo o nome do projeto em uma pasta. Dessa forma, com o projeto criado, podem ser criados os arquivos de cabeçalho (*headers*), aqueles com as implementações das funções (*sources*) e a *main*. Como os arquivos novos, por padrão, são adicionados na pasta *Source Files*, basta arrastá-los para as

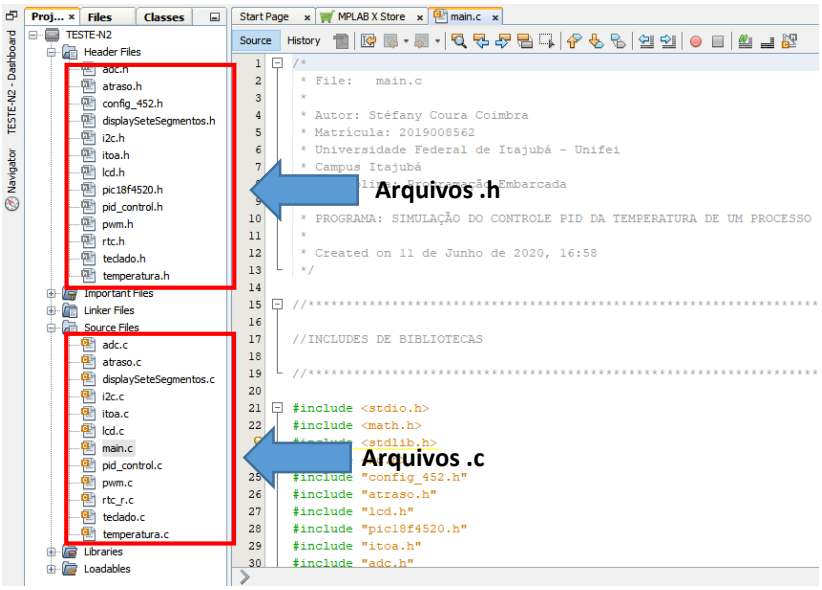


Figura 2 – Organização do projeto

pastas certas no próprio software. Para compilar o código o botão com um martelo pode ser pressionado ou ainda o com um martelo e uma vassoura. Para compilar o projeto, basta clicar com o botão direito do mouse no projeto e selecionar *Build*. Após a compilação dos códigos, um arquivo .hex é gerado e será utilizado na simulação. Problemas com o MPLAB X IDE foram solucionados a partir do download de uma versão anterior à mais nova disponibilizada pelo site oficial.

Passo 2: O Hardware

Depois de desenvolver os códigos no MPLAB X IDE, é hora de simular o projeto na ferramenta PICSimLab, utilizando o arquivo .hex gerado ao final do passo anterior. Para isso, é necessário configurar o software. É selecionada a placa PICenios, com o PIC18F4520, um clock de 8MHz, o LCD de 16X4 e ligada a chave do buzzer para que este componente funcione adequadamente. Houve algumas dificuldades relacionadas ao buzzer no projeto, principalmente conflitos com a porta PWM. Para solucioná-los, foi essencial possuir em mãos o *datasheet* do PIC18F4520 e identificado o problema relacionado à manipulação necessária do registrador TMR2ON pelo PWM. Dessa forma, testes foram feitos e por meio de alternância entre acionamento e não acionamento do buzzer, no final do projeto, foi garantido o sucesso. Além do *datasheet* do microcontrolador (https://ww1.microchip.com/downloads/en/DeviceDoc/39631E.pdf), através da aba *Help* do PICSimLab, foi possível conhecer e testar o funcionamento dos componentes da placa virtual. As funcionalidades das portas da placa são organizadas conforme a Figura 3 (disponibilizada também em https://lcgamboa.github.io/picsimlab/Features_Board_PICGenios.html). Para o projeto em questão, será necessário a multiplexação para os displays de 7 segmentos. Para valores maiores que 9 ou negativos, utilizando mais de um display, é alternado a ligação de cada um de forma rápida, de forma que permita a sensação de simultaneidade do valor mostrado.

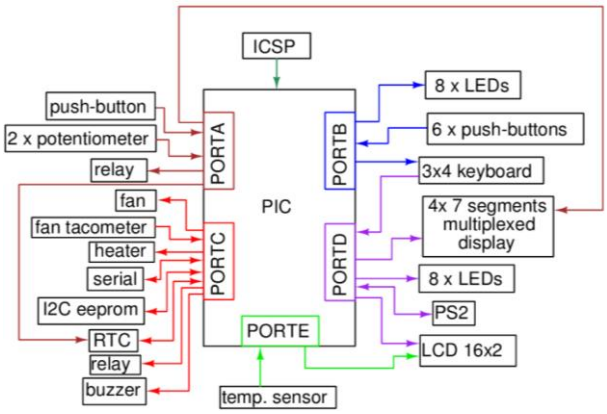


Figura 3 – Ligação dos componentes da placa ao PIC

Figura 4 – Configuração do PICSimLab

Passo 3: Funcionamento do Código

O código é dividido em partes, com comentários do funcionamento de cada função, de forma a ser organizado, de melhor entendimento e pelo projeto possuir fases.

A construção do código inicia-se com a declaração de todas as bibliotecas necessárias ao projeto, todas as variáveis gerais de todo o código e o protótipo das funções. Na ***main***, primeiramente, há uma inicialização do processo, com a mensagem "Bem-Vindo!" para o usuário, informações da data e hora do sistema no momento. No caso esse usuário, em um processo industrial, seria um técnico ou Engenheiro em Automação Industrial, por exemplo. Em seguida, é apresentada a mensagem da finalidade do processo (aparece no LCD "CONTROLE PID e GRANDEZA: TEMPERATURA), de forma a informar ao usuário qual grandeza está sendo medida no processo, já que o controle PID pode ser utilizado para diversas variáveis, em diferentes sistemas. A parte principal da ***main*** é a função ***menuGeral()***, que é um pré-menu do processo, com as opções de configuração e início do processo ou finalização, caso desejado.

No ***menuPrincipal()***, há as opções de configurações do Setpoint do processo, valor medido simulado da temperatura, das constantes Kp, Ki e Kd e visualização do processo. Do menu principal, há as subfunções para cada uma das opções. Com o direcionamento para a função ***menuSetpoint()***, através do pressionamento do botão RB3, o usuário é capaz de digitar um valor desejado para a temperatura do processo e voltar ao menu principal para as opções. Logo após do valor digitado, utilizando o teclado matricial, do Setpoint, ele é exibido no LCD para que haja confirmação do valor digitado pelo usuário e evite assumir valores equivocados para o processo. É importante notar que o pressionamento das teclas possui um certo *delay* e, por isso, é necessário atentar-se a isso ao simular o projeto.

No ***menuTemperatura(),*** o usuário pode regular a temperatura considerada medida do processo, o que seria equivalente ao recebimento desse valor de um instrumento sensor de temperatura. O incremento e decremento da temperatura é feito pelos botões RB0 e RB1, respectivamente, e o botão RB3 volta ao menu principal. A visualização do valor regulado para a temperatura pode ser vista de duas formas: utilizando o LCD e o display de sete segmentos, além de poder assumir valores negativos. É necessário frisar que houve uma dificuldade durante a realização do projeto na tentativa de assumir os valores decimais, devido à complexidade deste procedimento utilizando as ferramentas especificadas, portanto todos os valores do processo são inteiros. É feita, para a visualização da temperatura no display de sete segmentos, a multiplexação dos displays de 7 segmentos, escrevendo o algarismo equivalente ao tempo de modo repetitivo para criar a sensação de que todos os displays usados no momento estejam ligados juntos. Caso seja de interesse a mudança da passagem de tempo, é essencial observar o *efeito flicker* gerado quando o tempo de resposta dos displays é baixo e se apresentam piscando na placa.

Por meio do direcionamento para o ***menuConstantes()***, é possível alterar e setar os valores das constantes Kp, Ki e Kd, lembrando que o processo de controle de temperatura é baseado em malha fechada. Os valores são setados pelas teclas do teclado matricial. Caso o usuário não altere os valores das constantes, é assumido o padrão de Kp=1, Ki=Kd=0, pelo princípio de regulagem em um controlador PID, já que Ki e Kd estão relacionadas a um controle mais fino do processo. Em um sistema real, essa regulagem é feita, geralmente, de forma empírica a depender das variáveis de processo. Para a seleção dos valores das constantes e *Setpoint*, basta pressionar o botão #.

Finalmente, com o direcionamento para o ***menuProcesso()***, o processo de controle da temperatura pode ser visualizado. É utilizado o PWM (as implementações das funções relacionadas ao PWM estão contidas no arquivo *pwm.c*) para regular a potência do cooler e do heater e para acionamento do relé, utilizados no processo a depender dos valores de temperatura desejado e medido

pelo processo. No arquivo *pwm.c*, Há a inicialização dos registradores T2CON, CCP1CON e CCP2CON, responsáveis pelo uso do PWM no PIC18F4520, a função da definição de frequência do PWM e de *duty cycle*.

A função principal do projeto que realiza, de fato, o controle PID, é a *pid*, implementada no arquivo *pid_control.c*. Nela, há as variáveis e cálculos do *P(proporcional), D(derivativo) e I(integrativo)* e o controle final é dado pela soma das três variáveis.

Passo 4: Simulando o projeto com o PICSimLab

Para simular o projeto, é necessário abrir o arquivo *.hex* gerado pela compilação. Ele se encontra no caminho *Nome_do_projeto → dis → default → production*.

Na simulação, nos displays de sete segmentos, será visualizado o valor medido da temperatura no sistema (simulada por botões). O teclado e botões mencionados serão usados para o usuário digitar os valores das constantes Kp, Ki, Kd e do *Setpoint* do processo, além do acesso ao menu de configuração dos valores para a aplicação do PID e para simulação do valor do potenciômetro (valor medido da temperatura, no sistema analisado). O valor real da temperatura, que seria obtido no processo em si por um sensor da grandeza observada, será também simulado por teclas/botões. Os Leds serão parte do monitoramento do sistema. Cada um tem um papel: um é acionado caso o valor de temperatura seja menor que o *Setpoint*, outro é ligado caso o valor de temperatura seja maior que o *Setpoint* e um terceiro é ligado caso a medida seja, em algum momento, igual ao *Setpoint*. O cooler e heater representam atuadores no contexto do processo. O primeiro será ligado caso o valor medido da temperatura seja maior que aquele desejado para o processo e sua velocidade angular (ou o tempo em que permanecerá ligado) será regulada pelo PWM. O segundo, caso a temperatura real seja menor que aquela desejada. O relé representa um dos atuadores do processo e será acionado para aquecer o sistema de forma a aumentar a temperatura, indo de encontro ao *Setpoint*. Neste caso (em que o valor real da temperatura seja igual ao valor do setpoint), o *buzzer* é ligado como um sinal de alarme. Serão mostrados no LCD da placa os valores das constantes para alteração (menu de configuração), o valor do setpoint e o valor do PWM no momento.

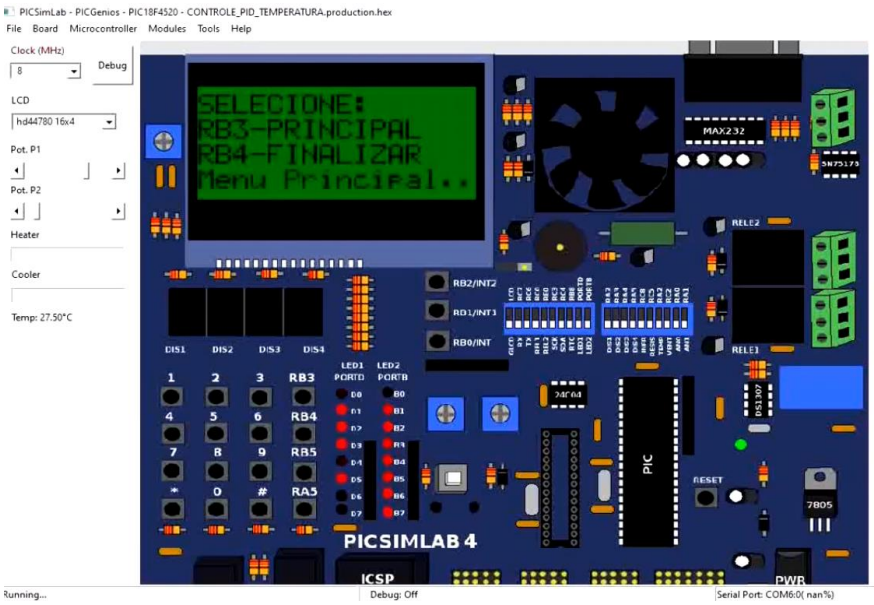


Figura 5 – Menu Geral do processo durante a simulação

Passo 5: Vídeo da explicação do projeto e disponibilização dos códigos

Para saber mais sobre o projeto, acesse:

https://www.youtube.com/watch?v=VYgokGqxafo

Para visualizar, alterar e contribuir com o projeto, acesse o GitHub com os códigos e documentos:

https://github.com/stefanycoimbra/Projeto-Simula-o_Controle_PID_Temperatura