Projeto Final - turma B

Projeto de sistemas embarcados

Cassio Gião Dezotti RA: 168988
Gustavo Parapugna Moraes RA: 174217

UNIVERSIDADE ESTADUAL DE CAMPINAS - UNICAMP

Campinas, 01 de agosto de 2020

SUMÁRIO

1. Resumo do projeto

O projeto proposta para esta disciplina consiste em um controlador de temperatura através do kit de desenvolvimento FRDM KL25 acoplado com a placa de periféricos MCLAB2. O grupo seguiu os requisitos de projeto fornecidos e inseriu outros, todos os requisitos funcionais e não funcionais estão lidos na tabela de requisitos.

A principal função do sistema é receber uma informação fornecida pelo usuário, que será um valor de temperatura ou um valor para cada ganho do sistema (Kp, Ki, Kd), e então a partir desses dados fazer o cálculo do duty cycle para saber qual a potência que será imposta no aquecedor, a fim de controlar a temperatura geral do sistema. Uma outra funcionalidade do sistema é a capacidade de se obter os valores atuais de temperatura e ganhos do sistema.

A definição dos valores pode ser realizada por botão do tipo *push button* ou via comunicação serial RS-232. O set por botão funciona através de uma interrupção externa em que o próprio usuário irá acionar o botão requisitando o acesso, já a comunicação via serial ocorrerá por meio de uma interface onde o usuário será capaz de escrever a mensagem pelo teclado gerando uma interrupção no sistema até que a definição do parâmetro desejado seja feita.

Já a leitura dos valores deve ser feita pela comunicação serial que lê os valores no sistema e retorna como resposta para o usuário.

2. Documentação do sistema

Por conta do tamanho dos diagramas criou-se um arquivo somente para essa seção com todos os diagramas juntos e contextualizados, o mesmo foi anexado no github juntamente com outros arquivos pedidos.

Pedimos desculpa pela dificuldade causada porém achamos melhor do que inseri-los aqui sem uma resolução adequada.

3. Manual de utilização

Após ligar o sistema o usuário deverá ver a temperatura atual exibida no display lcd, o valor inicial da temperatura desejada foi definido como a temperatura atual. Os valores de ganho foram definidos como 1.

A primeira coisa a ser feita é sintonizar o controlador. Recomenda-se isso porque o controle do aquecedor é realizado somente após a inserção dos primeiros valores de ganho, dessa forma caso a temperatura seja determinada anteriormente o sistema não irá conseguir realizar o controle correto e assim o aquecedor não estará com a potência necessária para representar a temperatura definida. Para isso recomendamos o método de Ziegler-Nichols de malha fechada, este método possui um sobresinal médio, alta robustez de ganho, tempo de subida médio e tempo de estabilização médio. Como nossa malha de controle é simples, acreditamos que este método será o melhor pelo fato de sua implementação ser simples e depender apenas da resposta ao degrau em malha fechada.

No passo a passo:

- Primeiro o usuário deve aplicar uma entrada de degrau unitário no sistema.
- Setar os ganhos Ki e Kd como zero.
- Aumentar gradativamente o ganho Kp até que seja obtido o limite de estabilidade (oscilações sustentadas).
- Quando esse limite for atingido, guardar o valor de Kp, esse será o valor do ganho final Ku.
- Medir o período entre as oscilações e guardar, esse será o valor do período último Pu.
- Com esses valores anotados calcular os valores de K, pelas seguintes fórmulas, Kp = 0.6*Ku, Ki = Pu/2, Kd = Pu/8.
- Com esses valores calculados, setar o controlador do sistema com esse ganhos.

3.1. Utilização dos botões

Para interface local, os 2 primeiros botões acionam os modos de definição do sistema.

Quando o segundo botão é pressionado o sistema entra em modo de definição dos ganhos do controlador, a partir de agora os botão assumem outras funções, o primeiro confirma o valor escolhido, o segundo acrescenta 0.1 ao valor escolhido, o terceiro acrescenta uma unidade ao valor escolhido e o quarto botão subtrai 0.1 no valor. O primeiro ganho a ser setado é o Kp, ele será exibido no display lcd até o usuário pressionar a tecla de confirma. Após a confirmação o usuário irá setar e ver o Ki no display. Após confirmar o usuário deverá ver e setar o Kd. O usuário deve pressionar o confirma mais uma vez, retornando o sistema para o estado inicial, onde é exibida a temperatura atual.

Quando o primeiro botão é pressionado o sistema entra em modo de definição da temperatura determinada (set point), o sistema irá exibir a temperatura determinada no display lcd. A partir de agora os botão assumem outras funções, o primeiro confirma o valor escolhido, o segundo acrescenta uma unidade ao valor escolhido, o terceiro acrescenta uma dezena ao valor escolhido e o quarto botão subtrai uma unidade no valor. O usuário deve setar a temperatura desejada e pressionar o botão confirma, logo após o sistema retorna ao estado inicial onde exibe a temperatura atual.

3.2. Utilização da comunicação serial RS-232

A definição dos valores via serial deve ser feita pela interface, onde o usuário deve seguir um protocolo de comunicação específico.

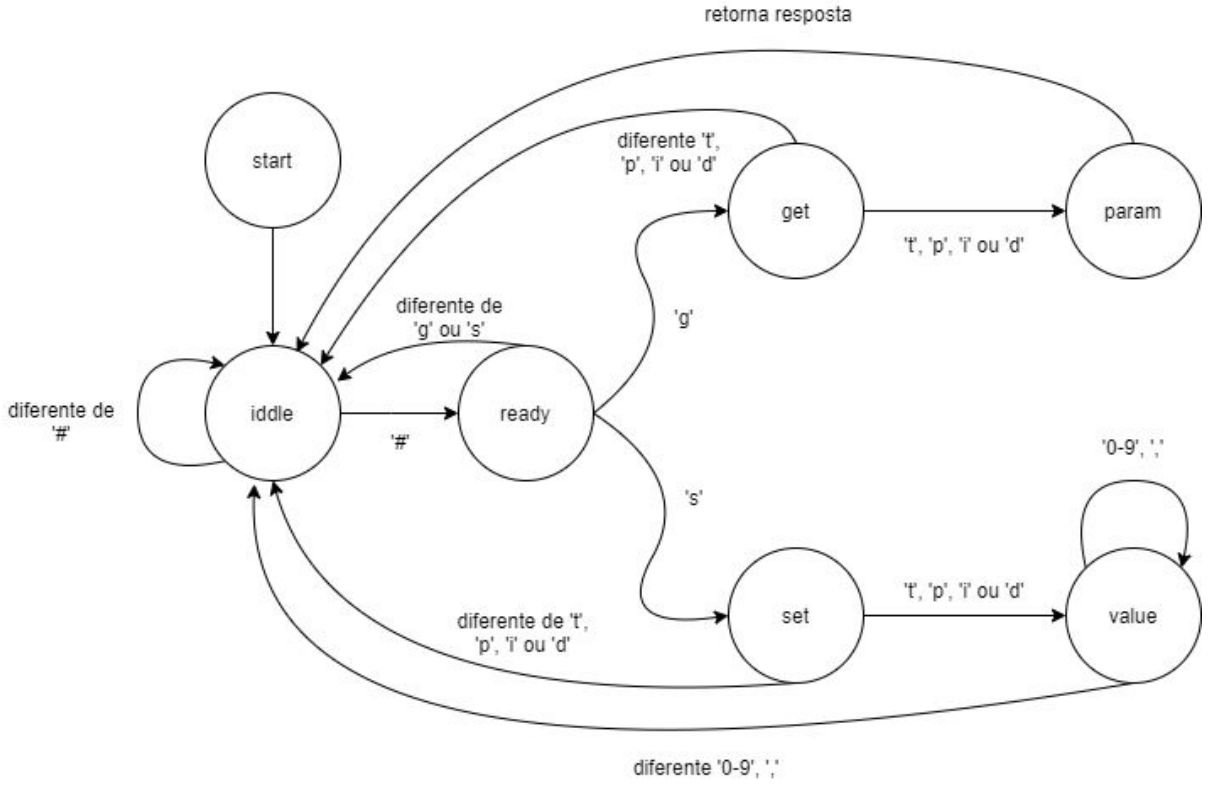

Figura 1 - Máquina de estados da comunicação via serial RS-232

Para o set de temperatura o usuário deve digitar o caractere “#”, em seguida o caracter “s” e depois o “t”. Para finalizar deve-se digitar o valor corresponde ao que se deseja determinar.

Para o set dos ganhos deve-se informar os mesmos passos iniciais de “#” e “s”, porém agora é necessário escrever “p”, “i” ou “d” para determinar qual o ganho escolhido e em seguida o valor desejado.

A leitura através da comunicação serial deve ser feita da seguinte maneira: escreve-se primeiramente os caracteres “#” seguido do “s” e depois deve-se inserir o caractere de get “g”, por fim pede-se para que seja enviado o caracter que representa o parâmetro desejado, sendo “t” para a temperatura, “p” para o ganho proporcional Kp, “i” para o ganho integral Ki e “d” para o ganho derivativo.

4. Problemas identificados e não resolvidos

Após a conclusão do projeto a dupla analisou os resultados e levantou os seguintes problemas não resolvidos:

- Um requisito do sistema é manter a temperatura máxima em 90º, limitamos o valor do duty cycle do aquecedor em 0.5, mas não conseguimos realizar testes para confirmar isso.
- Um requisito do sistema é aquecer o resistor o mais rápido possível, mas como não implementamos o controlador do cooler e mantivemos ele sempre em 50% de duty cycle, acreditamos que esse requisito não foi atendido.
- Um requisito do sistema é manter o overshoot máximo deve ser de 1°C, como é um requisito não funcional, não conseguimos verificar se está sendo atendido.
- A interface para utilização da UART no computador não foi desenvolvida, por falta de acesso a placa de testes, o que dificultou a validação da mesma.
- Para a interface local, tivemos dificuldade em como elaborar um fluxo de fácil entendimento ao usuário. Nossa solução foi utilizar duas interrupções diferente por botões, mas isso pode confundir o usuário. Uma possível solução seria exibir um passo a passo informando ao usuário as funções de cada botão no lcd, mas não foi implementada por falta de tempo.

A dupla também listou algumas dificuldades encontradas na elaboração do projeto:

- Dificuldade de entendimento da comunicação UART, como não conseguimos debugar o código foi difícil entender como as conversões de valores recebidos e enviados estava funcionando, por conta disso optamos por trabalhar

basicamente com variáveis globais, evitando a manipulação de parâmetros pelas funções.

- Verificar o funcionamento do PID, como não conseguimos debugar o código foi difícil entender se a sintonização do controlador e seu funcionamento estão corretas.
- Dificuldade de construção dos UML, como é uma metodologia que tivemos pouco contato, algumas dúvidas dificultaram a construção. Por exemplo o fato de nosso sistema utilizar o paradigma de programação estruturada e o diagrama de classes utilizar o paradigma de programação orientada a objetos. Outro exemplo, como construir a máquina de estados, sendo que nosso sistema não se divide em estados definidos.

5. Autoria dos códigos fornecidos

adc.c e h - Utilizamos o arquivo de autoria de dloubach, julioalvesMS, IagoAF e rbacurau.

board.h - utilizadas algumas referências do professor Rodrigo Bacurau, que foram fornecidas ao longo da disciplina.

communicationStateMachine.c e .h - baseada no código fornecido em sala. Função processamentoByte(unsigned char ucByte) foi utilizada do professor Rodrigo Bacurau. As funções para manipulação de union foram baseadas nos códigos fornecidos em sala.

fsl_debug_console.c - Utilizamos o arquivo de autoría de Freescale Semiconductor, Inc.

lcd.c e .h - Utilizamos função fornecida pelo professor Rodrigo Bacurau e adicionamos algumas funções.

lptmr.c e .h - Utilizamos o arquivo de autoria de dloubach.

lut_adc_3v3.c e .h - Utilizamos o arquivo de autoria de julioalvesMS & IagoAF & dloubach.

mcg.c e .h - Utilizamos o arquivo de autoria de dloubach.

pid.c e .h - Utilizamos o arquivo de autoria de julioalvesMS, IagoAF, rBacurau.

print_scan.c - Utilizamos o arquivo de autoria de MOTOROLA, INC. All Rights Reserved

print_scan.h - Utilizamos o arquivo de autoria de Freescale Semiconductor, Inc.

UART.c e .h - Utilizamos o arquivo de autoría de dloubach, rbacurau.

util.c e .h - Utilizamos o arquivo de autoría de dloubach.

6. Errata do código

Após a compilação e edição dos códigos fonte verificou-se que o arquivo *aquecedorECooler* encontrada no pdf referente aos códigos (Página 3) na função *definirDutyH* possui um erro importante, pois ele afeta diretamente a funcionalidade do sistema.

Na foto apresentada consta um *if* que verifica se o duty cycle recebido está entre 0.5 e 1, porém como este mesmo duty cycle recebido foi limitado a no máximo 0.5 percebemos que esta condição nunca seria satisfeita, assim nunca seria definido um novo valor de duty cycle no aquecedor.

Esse erro foi corrigido no código que foi submetido no github.