**Módulo 1: Tratamento de Dados Coletados de Sensor de Temperatura LM35**

Prof. Dr. José Jean-Paul Zanlucchi de Souza Tavares

|  |  |  |  |
| --- | --- | --- | --- |
| **Revisão** | **Data** | **Responsável** | **Observação** |
| 0 | 21/10/2020 | José Jean Tavares | Rev. Inicial |
| 1 | 01/03/2021 | José Jean Tavares | Atualização módulo 1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

ÍNDICE

Sumário

1. Introdução 5

2. Objetivos 6

3. Captura de dados por meio de plataformas microprocessadas 7

4. Análise Estatística dos Dados Coletados 10

6. Atividades 13

Índice de Ilustrações

Figura 1 – Exemplo de Diagrama de Montagem de um Arduino Uno com Sensor de Temperatura LM35 pela Porta Analógica A2 e Diagrama Esquemático de Ligação 7

Figura 2 – Esquema de um Filtro Ativo Passa-Baixa de 2ª Ordem do tipo Butterworth 11

Figura 3 – Atenuação e Defasagem de Filtro Ativo Passa-Baixa de 2ª Ordem do tipo Butterworth 12

# 1. Introdução

Há milênios a humanidade faz uso de diversos dispositivos como ar comprimido, fluidos, explosivos, energia mecânica, térmica, elétrica, magnética, dentre outras, para proporcionar uma qualidade de vida melhor a si mesma. Nesse sentido surgiram os sistemas automatizados. A base de todo e qualquer sistema automatizado está no uso de instrumentos (sensores e atuadores) capazes de se comunicar para atingir um fim específico, seja uma saída dentro de um determinado intervalo, seja em manter o sistema em um estado desejado.

Todavia, existe uma grande distância entre o modelo teórico de um sistema e do sistema físico. Isso se deve a aproximações e simplificações do modelo teórico e ao fato do sistema real apresentar diversos comportamentos desconhecidos. Isso faz com que qualquer instrumento físico apresente características singulares o que pressupõe erro nas medidas. Esses erros podem ser estáticos (erros constantes ou proporcionais à entrada) ou dinâmicos (variações temporais e de frequência da entrada a ser mensurada podem gerar maiores erros na medição). Uma dessas fontes das incertezas se deve a ruídos, quer interno ou externo ao sistema de medição, além de outros fatores. Existem também entradas modificadoras, como é o caso da variação de temperatura em determinados sensores. Para erros estáticos, faz-se uso, geralmente, de ferramentas estatísticas sobre o resultado das amostras.

Atualmente, os sistemas de medição vêm utilizando elementos eletrônicos e computacionais, o que faz com que seja importante fundamentar o estudante do assunto em como fazer uso dessas ferramentas com ênfase na captura e tratamento dos dados e erros adquiridos pelos instrumentos.

# 2. Objetivos

Esta atividade prática tem como objetivo familiarizar o aluno com o tratamento de dados advindos de sistemas computacionais e eletrônicos integrado a um sistema de medição de temperatura utilizando o sensor LM35.

Os objetivos específicos desse módulo são:

1. Conhecer as principais funções estatísticas do Octave para cálculo de erros;
2. Elaborar um relatório apresentando os principais resultados encontrados como média, desvio padrão, intervalo da amostra, assimetria e achatamento sem e com aplicação do critério de Chauvenet.

# 3. Captura de dados por meio de plataformas microprocessadas

O sensor LM35, a ser utilizado nesse caso, é um sensor de precisão que apresenta uma saída de tensão linear proporcional à temperatura em que ele se encontrar no momento, tendo em sua saída um sinal de 10mV para cada Grau Célsius de temperatura.

Esse sensor não necessita de qualquer calibração externa para fornecer com exatidão, valores temperatura com variações de ¼ºC ou até mesmo ¾ºC dentro da faixa de temperatura entre –55ºC e 150ºC.

A captura dos dados desse módulo prático ocorre através da programação do Arduino. Uma porta analógica deve ser definida como entrada e o programa principal deverá efetuar a leitura da mesma e sua publicação pelo canal seria do equipamento periodicamente.

O Arduino é uma placa de prototipagem rápida de 10 bits, sendo assim, toda coleta de dados tem um erro de quantização no valor de 1/210 da medida. Como a entrada analógica do arduino é de 0 a 5V, isso significa que 5V corresponde a temperatura máxima enquanto 0 corresponde a temperatura mínima.

O código abaixo mostra um exemplo da coleta do sinal pela porta analógica A2, sua publicação no canal serial com periodicidade de 500 em 500ms.

#include <String.h>

int sinal; // variável inteira que armazena o valor do sinal recebido

int temp;// variável inteira que armazena o valor da temperatura

void setup(){

Serial.begin(9600); //inicia canal serial do arduino com vel. de 9600bps

}

void loop(){

sinal = analogRead(A2);//sinal recebr o valor lido na porta analógica 2

temp = (5 \* sinal \* 100)/1024;// a divisão do sinal é por 2^10

Serial.println(temp);//publica o valor da variável temperatura no canal serial

delay(500); //espera 500 milissegundos para a próxima execução

}

Esse código pode ser aplicado a diversas montagens físicas, quer dizer, para inúmeros tipos de sensores analógicos, desde que os mesmos estejam conectados à entrada analógica 2 e seu sinal seja de 0 a 5V.

Um exemplo de montagem física pode ser vista na Figura 1, onde um sensor de temperatura do tipo LM35 foi conectado à porta analógica A2 de um Arduino Uno. É possível notar a conexão de alimentação do sensor pelas portas 5V e GND do Arduino.

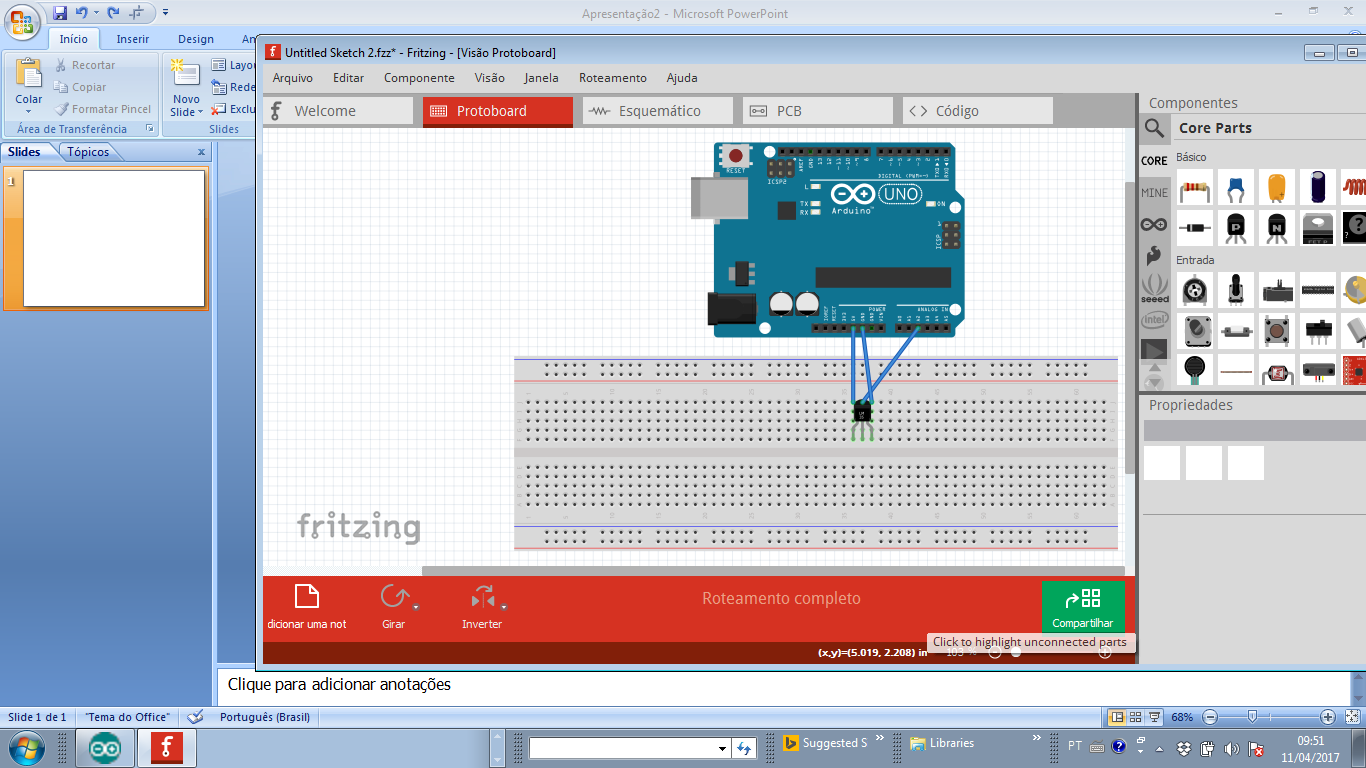
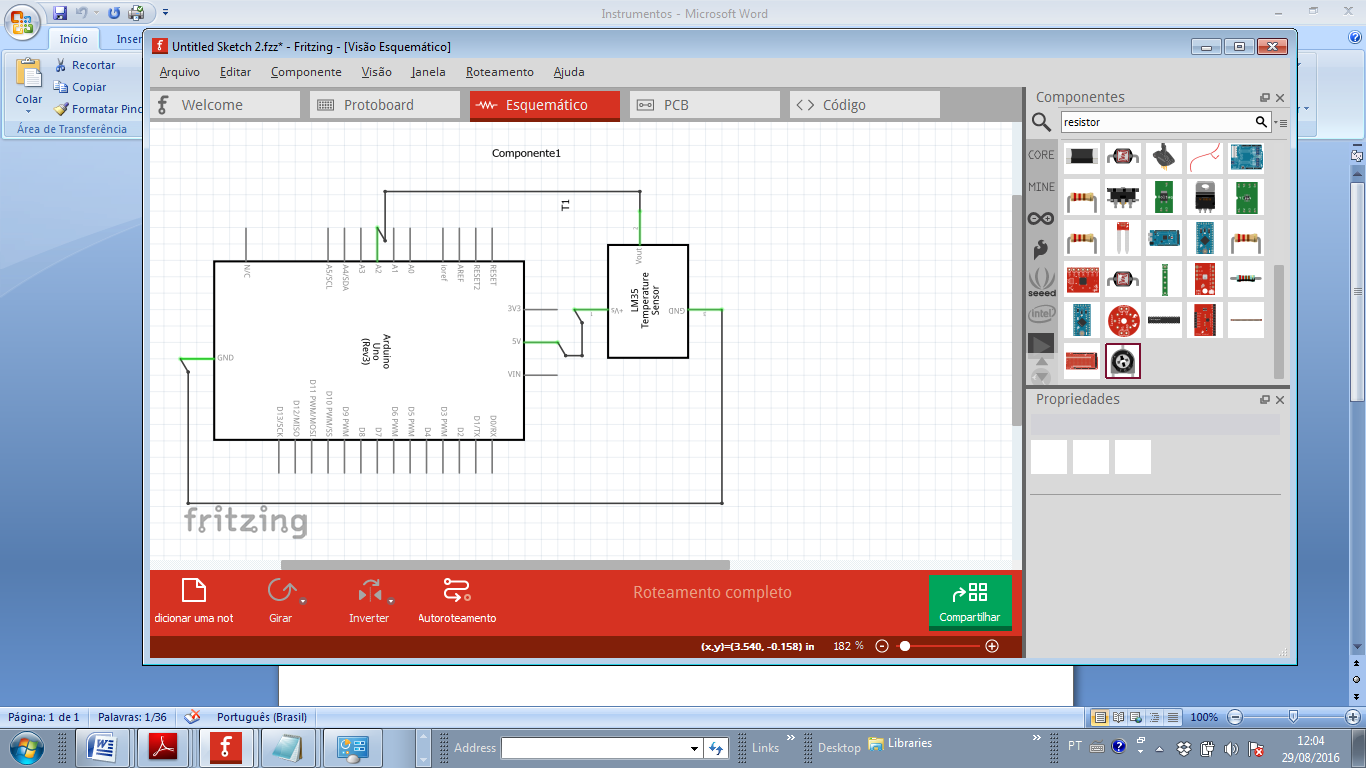
 

Figura 1 – Exemplo de Diagrama de Montagem de um Arduino Uno com Sensor de Temperatura LM35 pela Porta Analógica A2 e Diagrama Esquemático de Ligação

O Arduino é ligado pelo cabo USB com o computador, o que faz com que a porta serial do Arduino seja habilitada para comunicação, automaticamente. A comunicação só será realizada realmente se o comando “Serial.begin(*TaxaComuicação*)” for implementado no *setup* do código.

É importante avaliar qual o número da porta serial aberta no computador para comunicação com o Arduino através do “Painel de Controle” do *Windows*.

Para capturar os valores advindos da porta serial do Arduino é necessário um programa no Matlab que reconheça a porta serial utilizada. O código a ser utilizado pelo grupo está descrito abaixo. Note que a porta utilizada é a COM3. Esse valor deverá ser alterado conforme a porta de cada Computador atribuída ao Arduino. Ao final, o resultado é armazenado em um arquivo do tipo ‘txt’ e os dados são plotados graficamente.

s = serial('COM5'); %Usa a porta COM3 (colocar a porta referente a conexão do

%Arduino)

fopen(s); % Abertura da porta.

%ATENÇÃO: Só rode o programa se o Arduino não estiver sendo acessado pelo IDE!

residuo=0; %Variável que avalia resíduo vindo da porta de comunicação

N=input('Determine o número de dados a serem salvos: ');

%recebe o valor digitado pelo usuário no MATLAB

fprintf('\n')

fprintf('Aperte qualquer botão para iniciar');

fprintf('\n')

pause

pvet=zeros(1,N); % cria um vetor *pvet* com *N* posições contendo zero

% INÍCIO DO LAÇO DE CAPTURA DE *N* DADOS

for j=1:N % para cada leitura até N faça

p = []; % zera o valor de *p* para iniciar nova coleta de dados

flag=0; % Variável booleana que identifica se todo resíduo de comunicação já foi

% retirado... se 0, não....se 1, sim

% INÍCIO DO LAÇO PARA LIMPAR O CANAL DE COMUNICAÇÂO

while (s.BytesAvailable >0 && flag==0)

% Elimina o resíduo se a *flag* for zero e ainda houver *bytes* no canal de comunicação

residuo = fscanf(s);

end % FIM DO LAÇO PARA LIMPAR O CANAL DE COMUNICAÇÃO

flag=1; % Identifica canal de comunicação liberado para iniciar a coleta dos

%dados

% INÍCIO DO LAÇO PARA CAPTURAR OS DADOS DO CANAL DE COMUNICAÇÃO

while (s.BytesAvailable == 0)

% Enquanto não tiver *bytes* disponíveis na porta de comunicação

pause(0.05); % espera 50ms

end

while (s.BytesAvailable >0 && flag==1)

% Enquanto houver dados prontos para serem obtidos

p1 = fscanf(s); %*p1* recebe dados em formato *string*

p1 =str2num(p1); %*p1* transforma *string* em número

p=p1(1); %armazena resultado inteiro em *p*

flag=0; % encerra esse laço

end % FIM DO LAÇO PARA CAPTURAR OS DADOS DO CANAL DE COMUNICAÇÃO

pvet(j)=p; % Armazenamento do valor de *p* no vetor *pvet*, posição *j*

end % FIM DO LAÇO DE CAPTURA DE *N* DADOS

fclose(s); % Fecha a porta de comunicação

%ARMAZENAGEM DOS DADOS EM ARQUIVO TXT

fid = fopen('testeX.txt','wt') %abre arquivo testeX.txt por meio de *fid*

fprintf(fid,'%f\n',pvet) %transfere o vetor *pvet* para *fid*

fclose(fid) %fecha arquivo *fid*

fprintf('Dados salvos no arquivo Dados!!! ');

fprintf('\n')

%PLOTAGEM DOS DADOS EM FIGURA 1

n=1:1:N;

figure(1)

plot(n,pvet)

title('Dados Originais Coletados')

# 4. Análise Estatística dos Dados Coletados

Para efetuar a análise dos dados coletados é importante calcular média, desvio padrão, intervalo da amostra, assimetria e achatamento.

**Acrescente** ao código de captura de dados o código em Matlab a seguir para o cálculo dos valores estatísticos.

%Cálculo dos Valores estatísticos

disp('O valor da média é:')

pm=mean(pvet) %média da amostra

disp('O valor do desvio é:')

sp=std(pvet) %desvio da amostra

disp('O valor do intervalo da amostra originakl é:')

rp=range(pvet) %intervalo da amostra

disp('O valor da assimetria é:')

skp=skewness(pvet) %Skweness da amostra

disp('O valor do achatamento é:')

ktp=kurtosis(pvet) %Kurtosis

Após esse cálculo deve-se avaliar se há valores espúrios e retirá-los da amostra. Nesse caso, faz-se uso do Critério de Chauvenet que determina um intervalo de confiança estatístico máximo admissível e retira os valores amostrados que estiverem fora do intervalo. **O código de implementação em Matlab para o Critério de Chauvenet** está apresentado a seguir e **deverá ser copiado na sequência dos cálculos estatísticos**. Inicialmente se calcula o valor máximo admissível para depois avaliar quais são os valores discrepantes, armazenando o índice deles num vetor *indice()*. Se houver valor discrepante, o código remove-os e recalcula os valores de média, desvio padrão, intervalo da amostra, assimetria e achatamento com os dados após Chauvent.

% Critério de Chauvenet

DRo=norminv(1-1/(4\*N),0,1); %valor máximo admissível

pr=abs(pvet-pm)/sp;%calcula o desvio de cada medida

k=1

indice(k)=0

% LAÇO PARA IDENTIFICAR VALORES DISCREPANTES

for j1=1:N, % para N valores faça

if pr(j1)>=DRo % se o desvio é maior ou igual ao valor máximo

indice(k)=j1; % armazena o valor do índice j1 no vetor *indice()*

disp('o indice abaixo será retirado')

indice(k)

k=k+1;

end % FIM DO SE

end % FIM DO LAÇAO DE IDENTIFICAÇÃO

pch=pvet;%copia *pvet* para *pch*

if indice(1)~=0 %se houver algum índice para ser retirado

pch(indice)=[];%remove os valores com pr>DRo

end

Nc=length(pch);%tamanho dos dados após Chauvenet

% APRESENTAÇÃO DOS RESULTADOS APÓS CHAUVENET

nc=1:1:Nc;

figure(2), plot(n,pvet,nc,pch) %Plotar os dados brutos e modificados

title('Dados Originais e Corrigidos por Chauvenet')

legend ('Dados brutos','Dados com Chauvenet')

disp('O valor da média com Chauvenet é:')

pmc=mean(pch)

spc=std(pch)

disp('O valor da média é:')

rpc=range(pch)

skpc=skewness(pch)

ktpc=kurtosis(pch)

Depois de aplicar esse método é possível calcular o Intervalo de Confiança da amostra tratada. **Adicione o programa abaixo no Matlab para o cálculo do Intervalo de Confiança**.

% Cálculo do Intervalo de Confiança da Amostra Tratada

gl=Nc-1;%Calcula o número de graus de liberdade da amostra

alfa=0.95 %Define o nível de confiança estatístico

desv=tinv((1+alfa)/2,gl)\*spc/sqrt(Nc)%Calcula o valor do desvio do intervalo de

% confiança

int\_inf=pmc-desv % Calcula o limite inferior do Intervalo de Confiança

int\_sup=pmc+desv % Calcula o limite superior do Intervalo de Confiança

Falta apenas gerar o gráfico do histograma e frequência acumulada da amostra. **Então, copie e cole essa parte do código no Matlab.**

% Histograma e frequência acumulada

classes=floor(1+log10(Nc)/log10(2)); %Define número de classes

figure, hist(pch,classes) %Gera e plota o histograma

title('Histograma')

[no,x]=hist(pch,classes); %número de ocorrências e centros de cada classe

deltak=diff(x); %amplitude de cada classe

% LAÇO PARA CALCULAR VALORES ACUMULADOS

nacum(1)=no(1);

for j2=1:classes-1,

nacum(j2+1)=nacum(j2)+no(j2+1);

end % FIM DO LAÇAO DE VALORES ACUMULADOS

figure, plot(x,nacum)

title('Função Acumulada')

# 5. Atividades

**Relatório (4 Pontos)**

Com base nos dados coletados e enviado pelo professor:

a) Realize a análise estatística da amostra dos dados originais do sensor (1 pontos).

b) Aplique o critério de Chauvenet para a amostra e compare os resultados estatísticos com e sem a aplicação desse critério. (1 pontos)

c) Apresente o histograma das amostras e o gráfico correspondente a função acumulada. (1 pontos)

d) Implemente o teste do Quiquadrado e apresente o resultado para a amostra, informando se o mesmo adere a uma fdp normal com 95% de probabilidade. (1 pontos)