Exercícios 15, 16 3 17

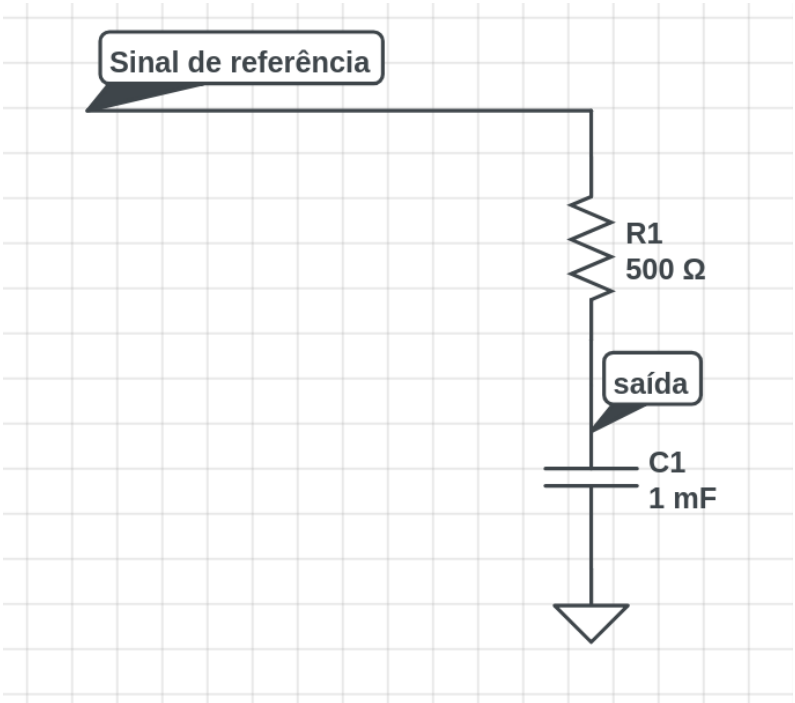
PROJETO E IMPLEMENTAÇÃO DE UM CONTROLE DIGITAL.

Nesse exercício você passará por todas as etapas envolvidas no projeto e implementação de um controle digital básico. Os slides publicados em blackboard podem seguir como guia para a elaboração do controle.

Valor: 7 pontos - Controlar a tensão no capacitor do circuito RC de modo a atender ao requisito de boa resposta ao sinal de referência:

Forma senoidal.
Frequência de 5Hz.
Offset 2,5V.
Amplitude de 200mv.

O circuito deve ser:



O sinal de referência deve ser gerado com o Analog Discovery. O controle deve ser implementado em arduino com uma taxa de execução de 1kHz.

A saída do circuito controlado deve ter ganho maior que 70% e defasagem menor que 30 graus para a entrada aplicada.

Valor – 3 pontos. Tente elaborar uma interrupção 20 vezes mais rápida que a interrupção principal que roda o controle (se necessário, pode diminuir a velocidade da interrupção 1, recalculando a planta e refazendo o controle para o novo valor de Ts).

Utilize a interrupção 2 para gerar uma média dos sinais de saida e referência a ser usada na interrupção principal.

Aumente a resolução do PWM através de um “dither”

Contents

Função de transferência em malha aberta:

$$G(s) = \frac{2}{s+2}$$

$$G_{discretizada}(s) = \frac{0.001998}{z - 0{,}998}$$

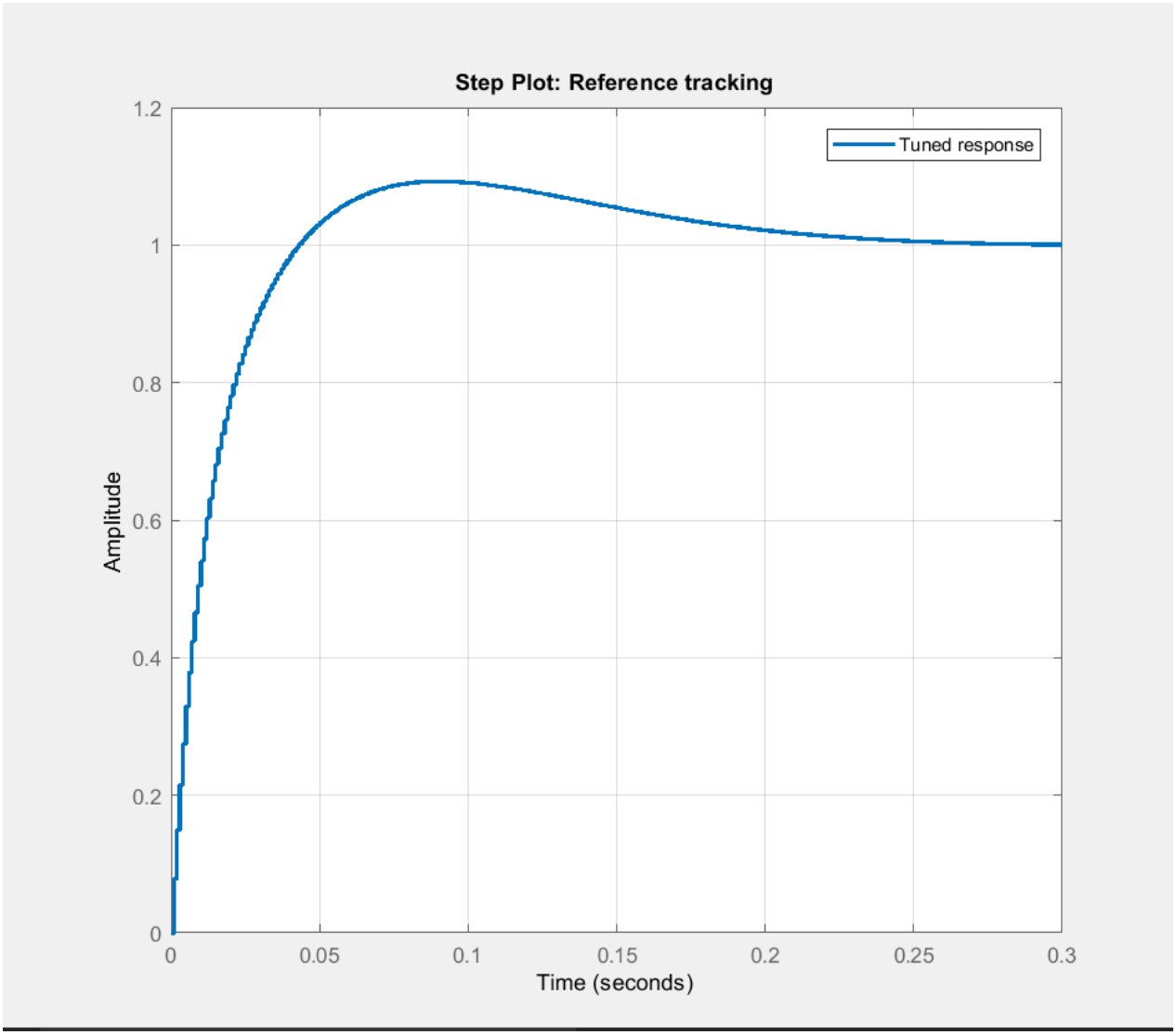
Controle Obtido:

$$G_{controlador}(s) = \frac{39{,}25z^2 - 76{,}33z + 37{,}1}{z^2 - 1{,}935 + 0{,}9355}$$

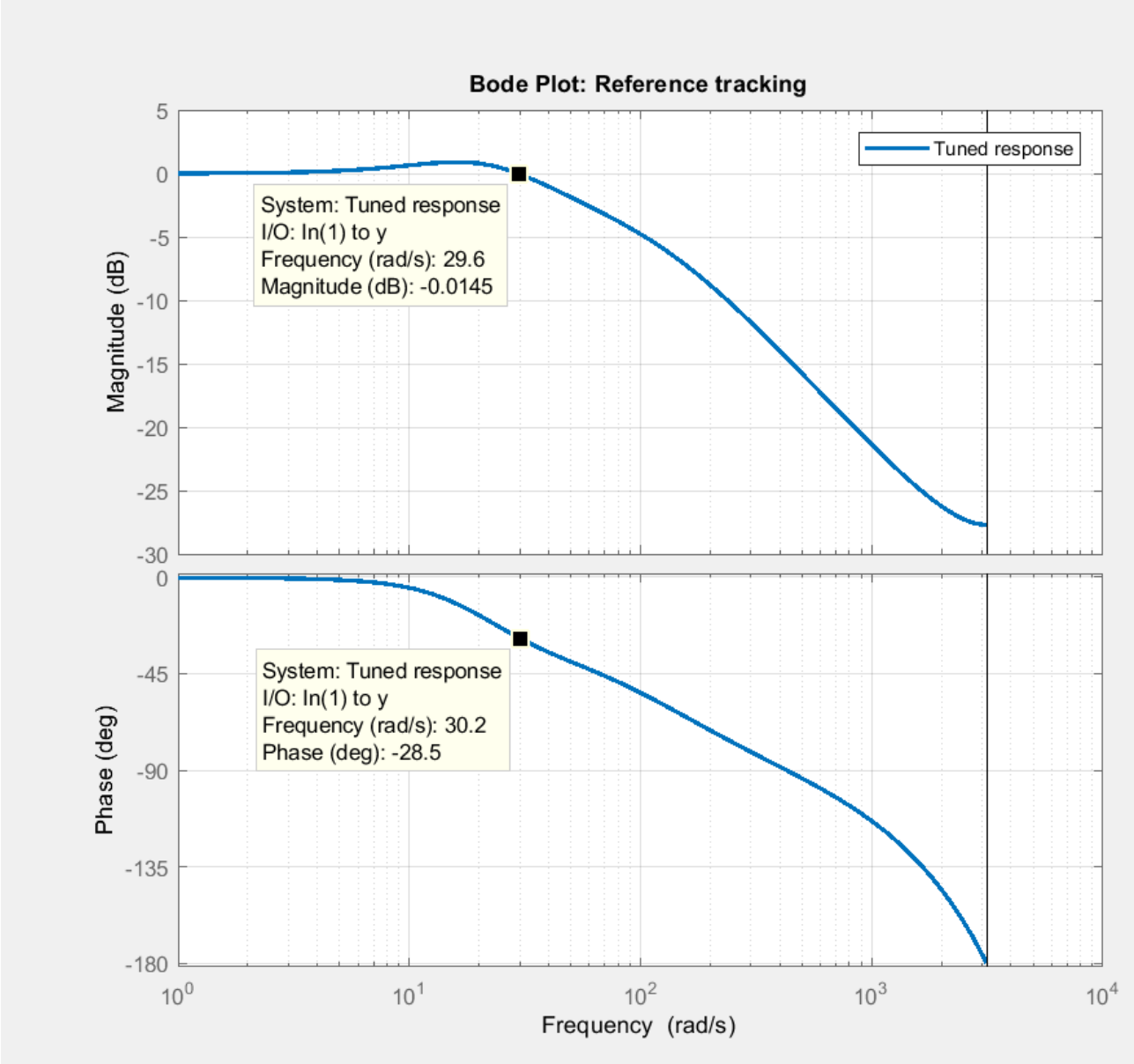
Sample time: 0.001 seconds

Discrete-time transfer function.

Step do sistema



Bode Plot


Bode Plot: Reference tracking
Tuned response
System: Tuned response
I/O: In(1) to y
Frequency (rad/s): 29.6
Magnitude (dB): -0.0145
Magnitude (dB)
5
0
-5
-10
-15
-20
-25
-30
System: Tuned response
I/O: In(1) to y
Frequency (rad/s): 30.2
Phase (deg): -28.5
Phase (deg)
0
-45
-90
-135
-180
10^0
10^1
10^2
10^3
10^4
Frequency (rad/s)

Resultados

Sem controle:

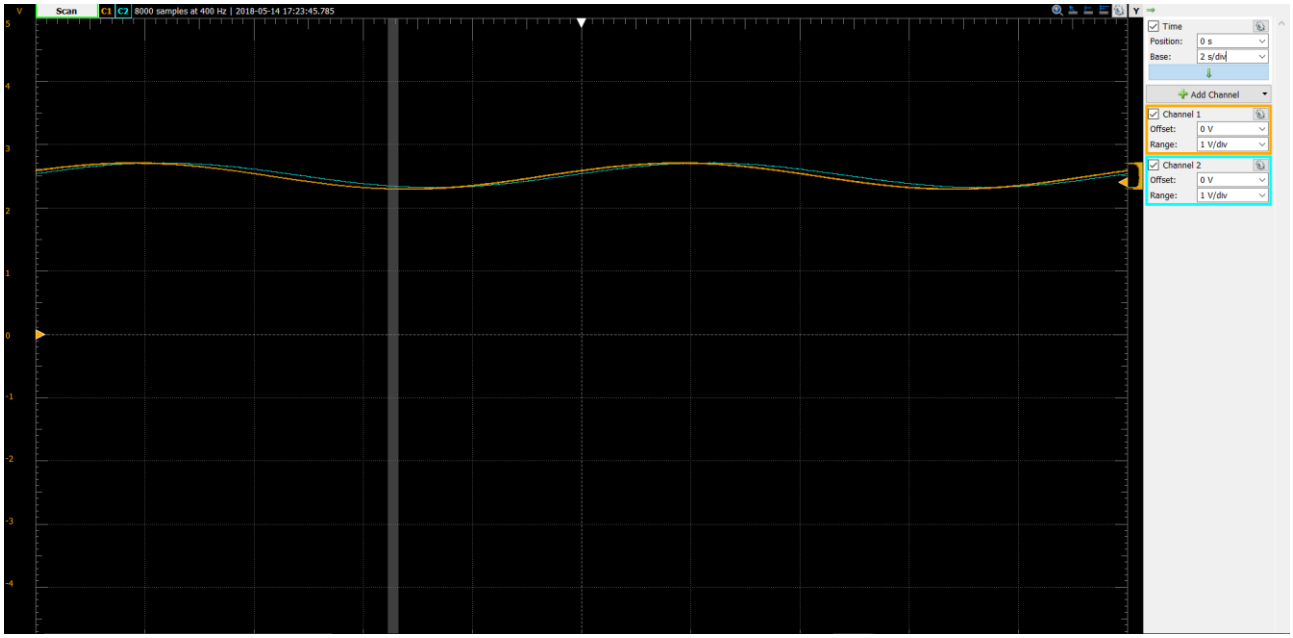


Figura 1 - 100mHz

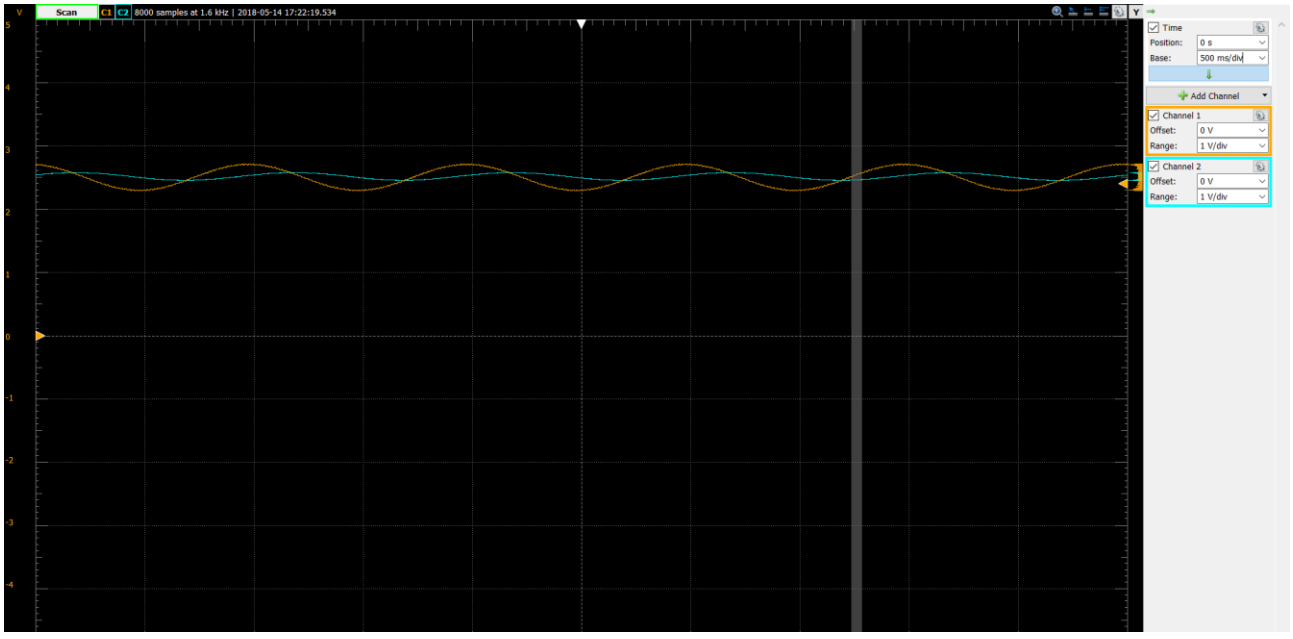


Figura 2 - 1Hz

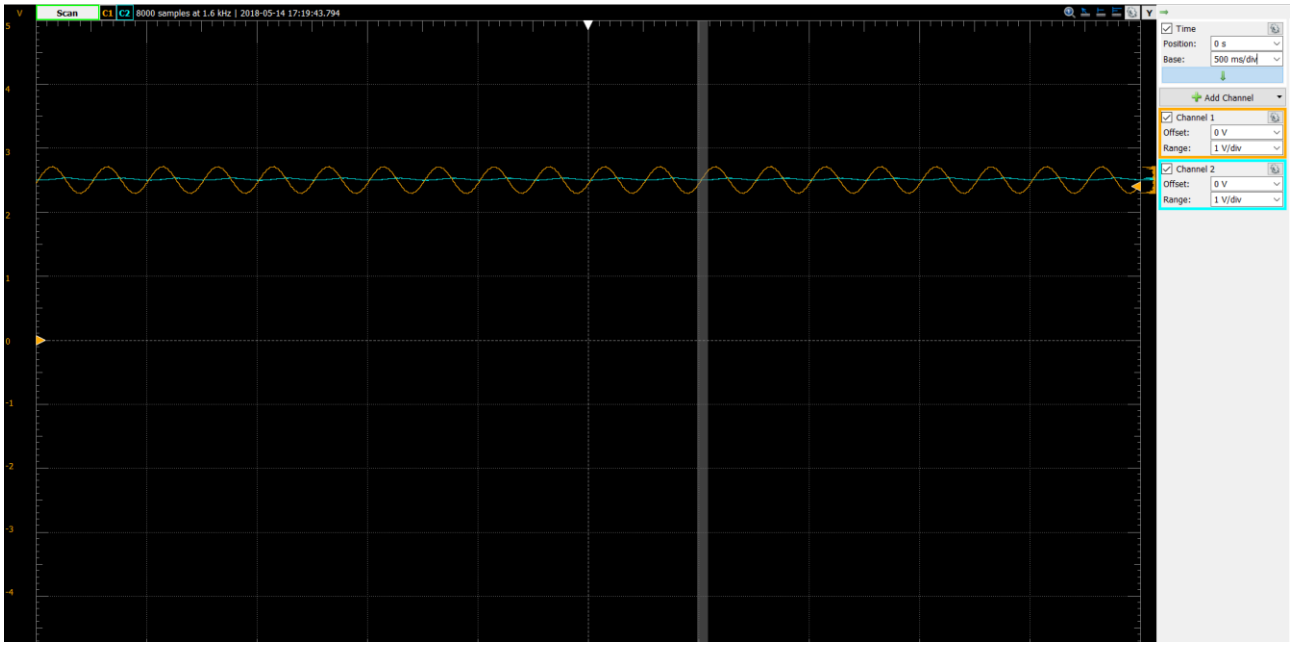


Figura 3 - 4Hz

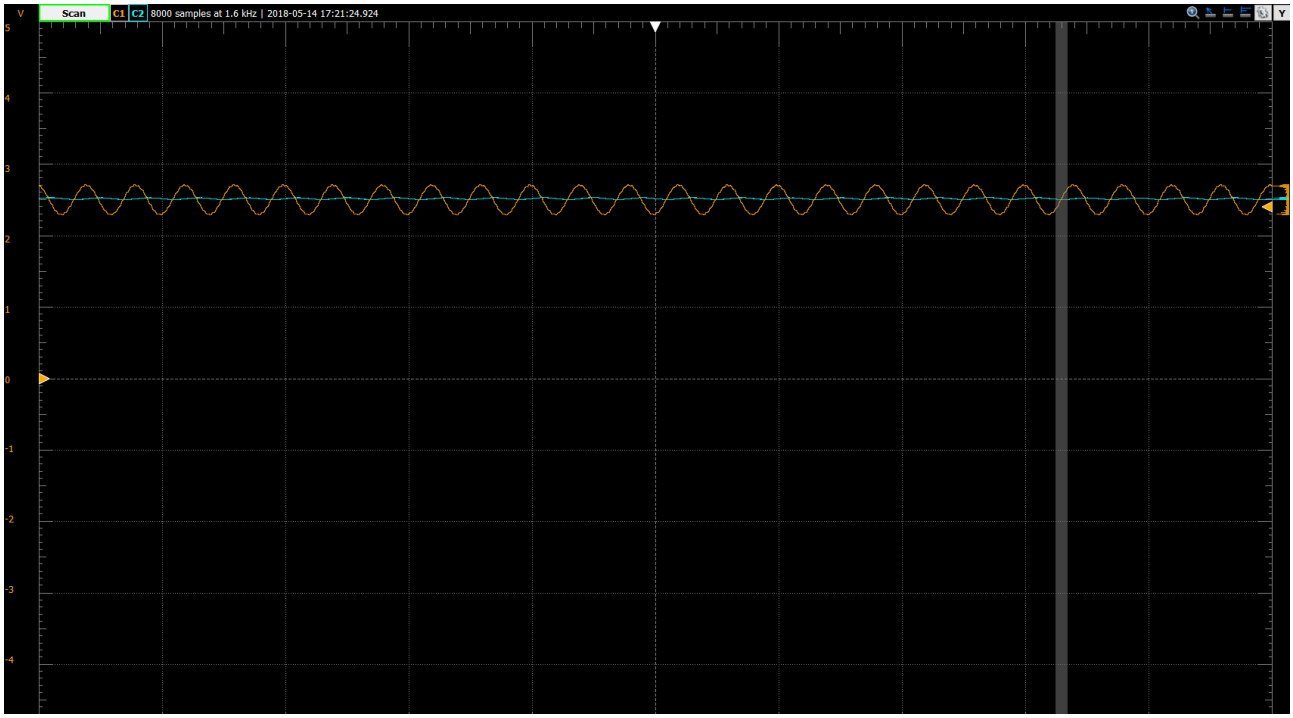


Figura 4 - 5Hz

Com controle

Com o prints

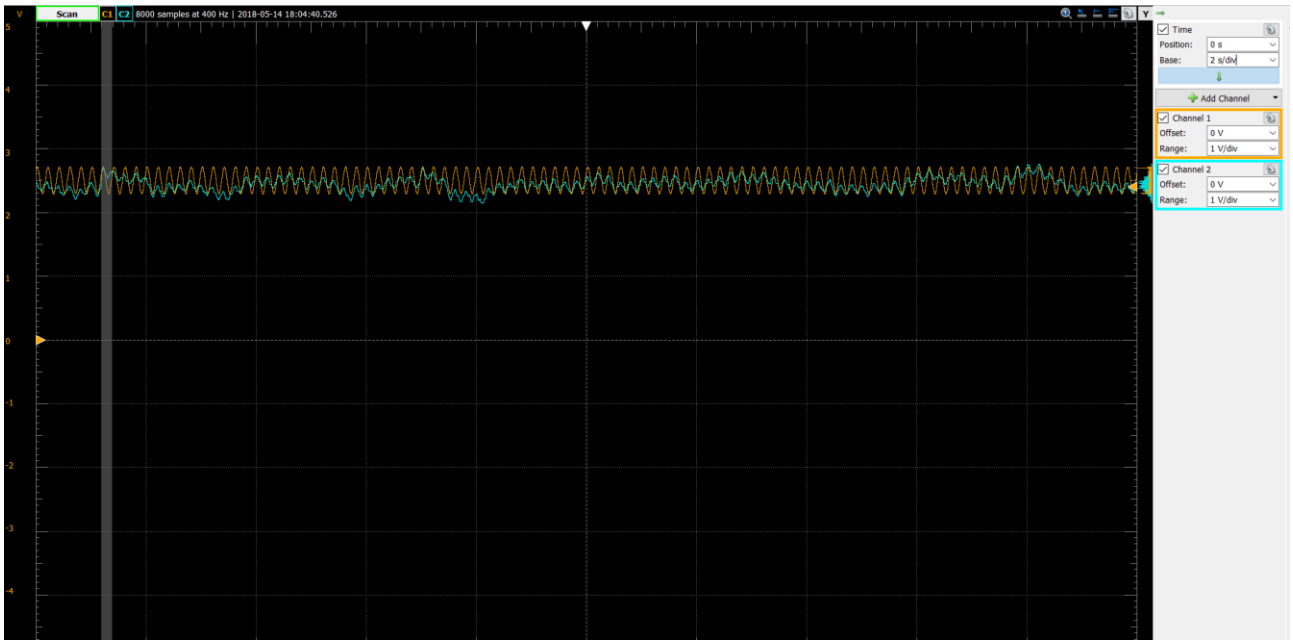


Figura 5 - 5Hz

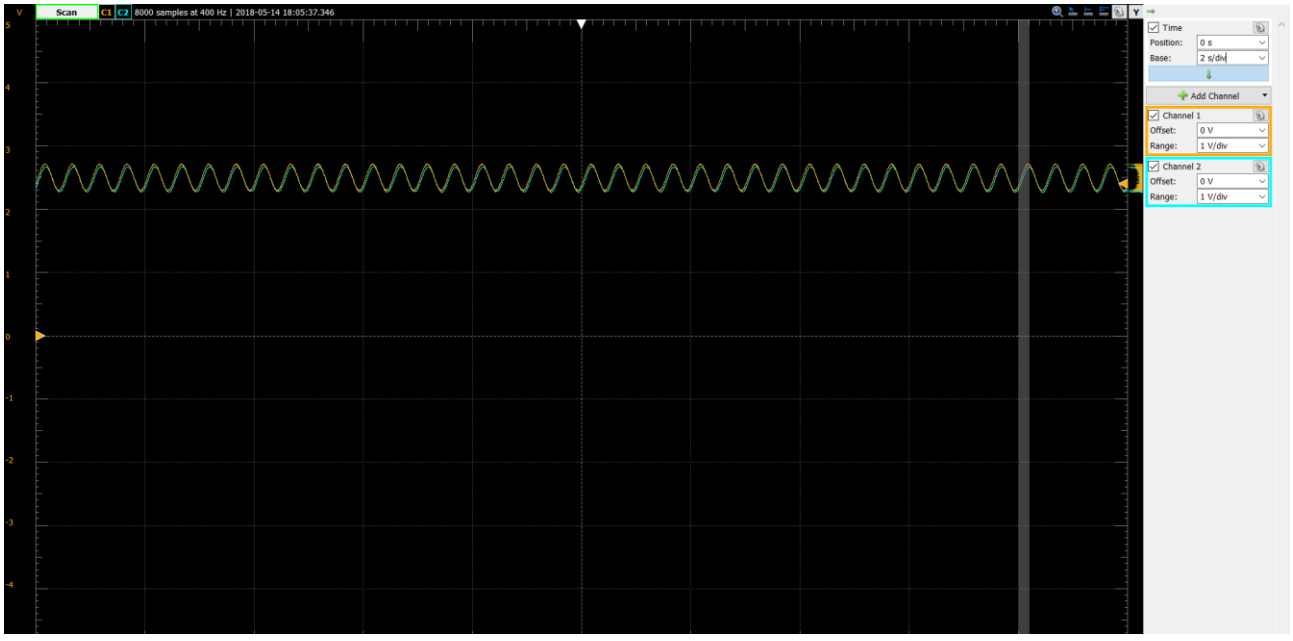


Figura 6 - 2Hz

Retirando os Printsln do código

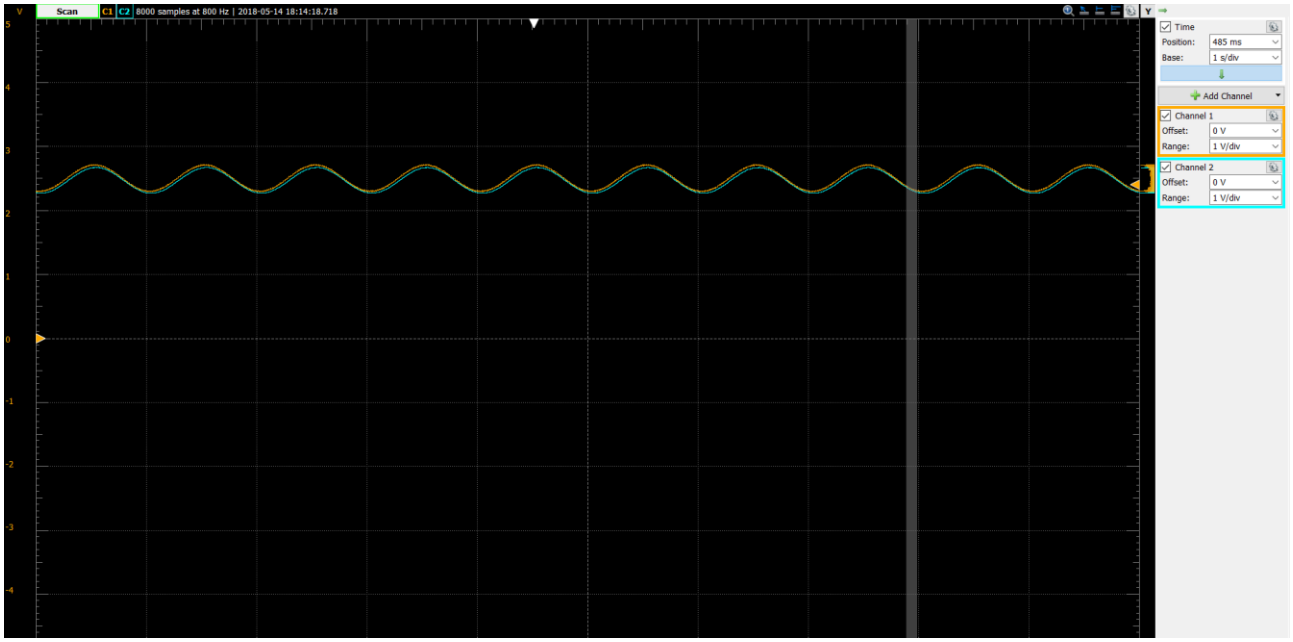


Figura 7 - 1Hz

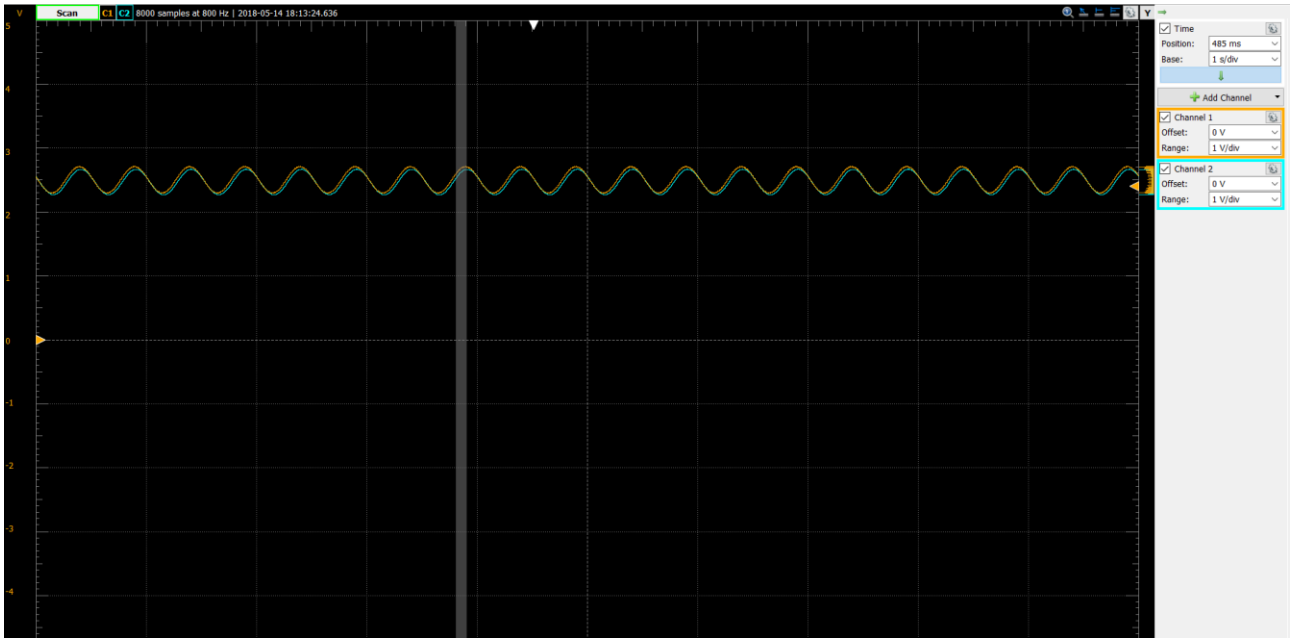


Figura 8 - 2Hz

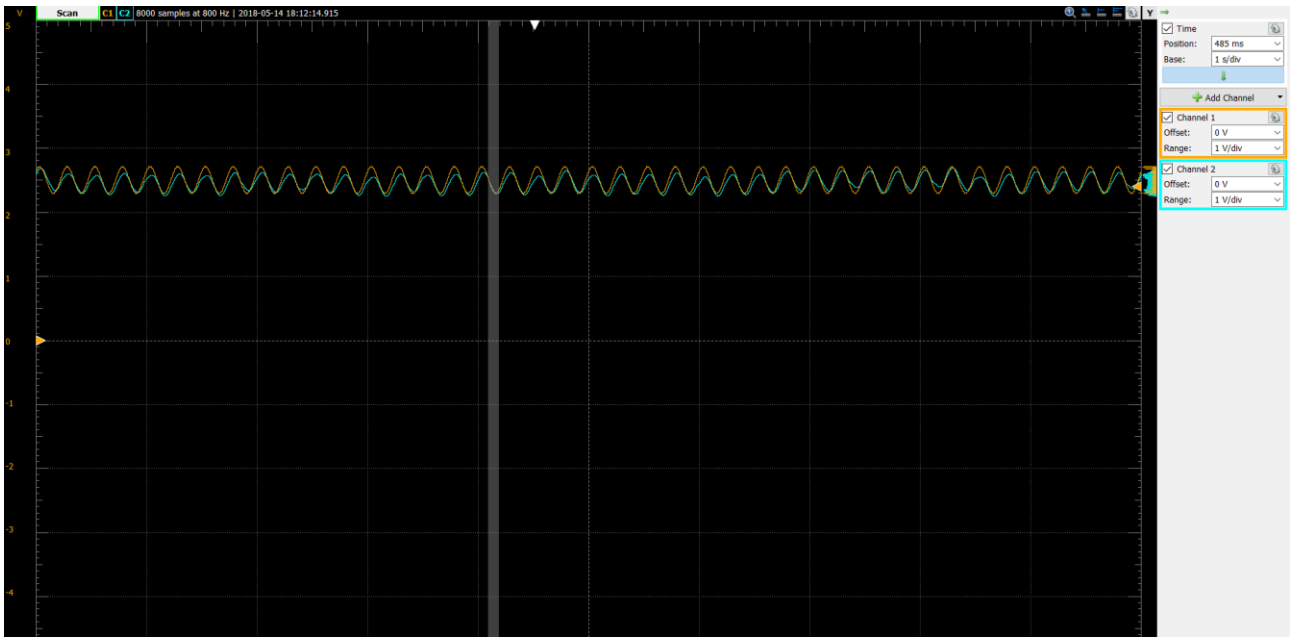


Figura 9 - 4Hz

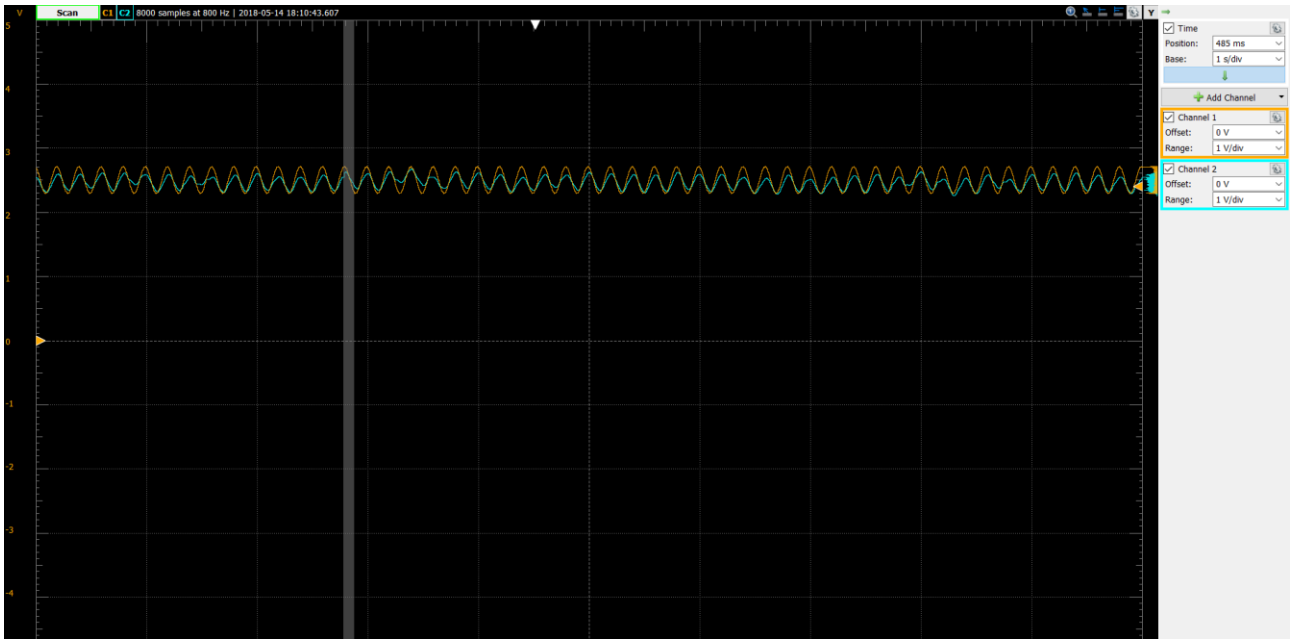


Figura 10 - 5Hz

Codigos

Código Matlab

```
function identifica_modelo = identifica_modelo(N)

%------------------------------le o arquivo--------------------------------
filename = 'C:\Users\Pedro Casella.DESKTOP-
T4HRM18.000\Desktop\Exercicio_12\SinalRbs_Bom.txt';
delimiter = ',';
formatSpec = '%f%f%f%f%[^\n\r]'; %import 4 colunas
%{
Coluna 1 e 3 são tempo; coluna 2 = sinal de entrada; coluna 4 = sinal de
saida
%}

fileID = fopen(filename,'r');
dataArray = textscan(fileID, formatSpec, 'Delimiter', delimiter, 'TextType',
'string', 'EmptyValue', NaN,  'ReturnOnError', false);
fclose(fileID);
SinalRbsBom = [dataArray{1:end-1}];
clearvars filename delimiter formatSpec fileID dataArray ans;
%----------------------------------------------------------------------------


%------------------------------Plot Entrada X Saida------------------------
entrada = SinalRbsBom(:,4); %entrada U
time = SinalRbsBom(:,1); %tempo
posicao = SinalRbsBom(:,2); % saida Y

figure(1) % Posição X Entrada
plot(time,posicao,'r',time,entrada,'b')
title('gráfico da entrada e saida')
legend('y - Saída','u - Entrada')
xlabel('time') % x-axis label
ylabel('Amplitude') % y-axis label

%----------------------------Obtendo vetor Velocidade----------------------

Ts = 0.002;
dposicao_dt=zeros(7501);
dposicao_dt=diff(posicao,1)/0.002;
dposicao_dt(7501) = dposicao_dt(7500);

figure(2) % Posição X Entrada
plot(time,dposicao_dt,'r',time,entrada,'b')
title('gráfico da entrada e saida')
legend('y - Saída','u - Entrada')
xlabel('time') % x-axis label
ylabel('Amplitude') % y-axis label

%-----------------------Escolhendo a ordem do modelo-----------------------

dposicao_dt = dposicao_dt(N:end); %ignora N pontos da entrada
entrada = entrada(N:end); %entrada U
DATA = iddata(dposicao_dt,entrada,Ts);

%na = ordem poli entrada
%nb = ordem poli saida
%nk = começa explicar a partir de que instante passado a entrada (polinomio B)

for i=1:5
```

```
    for j=1:5
        modeloIdentificado=arx(DATA,[i j 1 ]);
        AICarx(i,j)=modeloIdentificado.Report.FIT.AIC;
    end
end

for i=1:5
    for j=1:5
        if AICarx(i,j) == min(min(AICarx));
            na=i
            nb=j
    end
    end
end

nk = 1;

modeloIdentificado = arx(DATA,[na,nb,nk]); %identificação A(z) e B(z)

fprintf('FPE:')
FPE = modeloIdentificado.report.fit.FPE
fprintf('AIC:')
AIC = modeloIdentificado.report.fit.AIC


%----------------Criando a função transferencia e Simulando----------------

%função de transferencia representativa da velocidade
fprintf('Função de transferencia discretizada:')
Gd = tf(modeloIdentificado)
fprintf('Função de transferencia continua:')
Gv = d2c(Gd);
simplify(Gv)
Gi = tf([1],[1 0]);

fprintf('Função de transferencia da posição:')
Gs = Gi*Gv
simplify(Gs)

%plotagem dos polos e zeros do sistema
pzmap(Gs)
rlocus(Gs)

figure(4)
step(Gd,'c*')
grid on
figure(5)
step(Gv,'r')
grid on
figure(6)
step(Gs,'b')
grid on




%----------------- levantando controlador --------------------------------

R = 0.5*10^(3); %R = 1Mohns
C = 1*10^(-3); %C =  1uF

%função de transferencia (continua)
num = [1/(R*C)];
```

```
den = [1 (1/(R*C))];
G = tf(num,den)

%função de transferencia (discreta)
G_d = c2d(G,0.001)

%projeto controlador
pidTuner(G_d)
```

Código arduino

```
#include <TimerOne.h>

//declaração da variaveis
float erro=0;
float erro_1=0;
float erro_2=0;

float saida=0;
float saida_1=0;
float saida_2=0;

float set_point=0;


//declaração portas
int porta_pwm = 6;
int port_analog_discovery = A1;
int port_saida = A0;


void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);

  // pwm mais rapidos
   TCCR0B = TCCR0B & B11111000 | B00000001;
   TCCR2B = TCCR2B & B11111000 | B00000001;
   //analogWriteResolution(12); só para o DUE
   pinMode(6, OUTPUT);

  //timer
  Timer1.initialize(1000); //Inicializa o Timer e configura para um período de 0.001s
  Timer1.attachInterrupt(callback); //garante que a função demore 0.001s

}


void callback()
{
  //leitura set_point
  set_point = analogRead(port_analog_discovery);
  set_point = set_point/1023 * 5;
  Serial.println("Leitura Analog Discovery");
  Serial.println(set_point);

  saida = analogRead(port_saida);
  saida = saida/1023 * 5;
  Serial.println("Leitura Saida");
  Serial.println(saida);
```

```
 Serial.println("Leitura erro");
 Serial.println(erro);

 //setup dos erros
 erro_2 = erro_1;
 erro_1 = erro;

 erro = set_point - saida;


 //setup das saidas
 saida_2 = saida_1;
 saida_1 = saida;

 //função discretizada (entradas passadas e saidas passadas) - Saida = entrada G(S)
 //saida= 46.04*erro - 86.31*erro_1 + 40.33*erro_2 + 1.927*saida_1 - 0.9273*saida_2;
 //saida = erro*20.24 - erro_1 * 40.08 + erro_2 * 19.84 + saida_1 * 1.979 - saida_2 * 0.979;
 saida = erro*34.5118 - erro_1 * 66.6139 + erro_2 * 32.1283 + saida_1 * 1.9208 - saida_2 * 0.9208;
 Serial.println(saida);
 //saida = set_point;
 //saida = constrain(saida,0,255);

 saida = saida/5 *255;


  if(saida > 255 )
  {
   saida = 5;
   saida = 2.5 * 255 / 5;
  }
  if(saida <0 )
  {
   saida = 0;
   saida = 2.5 * 255 / 5;
  }




 //PWM
 analogWrite(porta_pwm, saida);
 //Serial.println("Saida Controlador");
 //Serial.println(saida);



 /*
 analogWrite(porta_pwm, 50);
 Serial.println("Saida Controlador");
 Serial.println(int(saida));
 */

}



void loop() {
  // put your main code here, to run repeatedly:

}
```